# Mathematical Foundations of Computer Networking

*S. Keshav*

*School of Computer Science*

*University of Waterloo*

keshav@uwaterloo.ca

Draft Version 1 - October 9 2009

# CHAPTER 1 A Short Introduction to Probability

This chapter presents a quick refresher course on probability.

## *1.1 Probability*

Intuitively, probability measures the degree of uncertainty about whether an *event* will occur. Given a set of potential events, called the *sample space*, the probability of an event chosen from this space is a real number between 0 and 1, where 1 represents that the event will surely occur, 0 represents that it almost surely will not occur (we need the 'almost' to deal with events whose outcome is a real number chosen from some interval: the probability of any specific real number occurring is zero, yet the probability that *some* real number in the interval is chosen is 1), and intermediate values reflect the degree to which one is confident that the event will or will not occur.

**Example 1: (Sample space and events)**

Imagine rolling a six-faced die numbered 1 through 6. Here, the sample space is the set of integers $\{1,2,3,4,5,6\}$. If the die is fair, the probability of any element of this set is 1/6.

Similarly, we could define the sample space of weather events for a particular day to be {rain, no rain}, and the probability that it will rain on that day could be 40% or 0.4.

[]

### 1.1.1 Axioms of probability

We denote events by *E* and the probability of an event by *P(E)*. The set of possible events, that is, the sample space, is denoted *S*. We assume that we can associate a real number *P(E)* with every element *E* in *S.*

The three axioms of probability, due to Kolmogorov, are:

1. $0 \le P(E) \le 1$ , that is, the probability of an event lies between 0 and 1.

2. *P(S) = 1*, that is, it is certain that at least some event in *S* will occur. This is also sometimes stated as *P(true) = 1* and *P(false) = 0*.
3. Two events $E_i$ and $E_j$ in *S* are *mutually exclusive* if $P(E_i\ E_j) = 0$, that is only one of the two may occur simultaneously. Given a sequence of mutually exclusive events $E_1, E_2,...$

$$P\left(\bigcup_{i=1}^{\infty} E_i\right) = \sum_{i=1}^{\infty} P(E_i) \tag{EQ 1}$$

That is, the probability that any *one* of the events in the sequence occurs is the sum of their individual probabilities.

Axiom 3 is defined for an infinite sequence of events. For any finite sequence of *n* mutually exclusive events, we can state the equivalent axiom as:

$$P\left(\bigcup_{i=1}^{n} E_i\right) = \sum_{i=1}^{n} P(E_i) \tag{EQ 2}$$

An alternative form of Axiom 3 is:

$$P(E_1 \vee E_2) = P(E_1) + P(E_2) - P(E_1 \wedge E_2) \tag{EQ 3}$$

### 1.1.2 Subjective and objective probability

The axiomatic approach is indifferent as to *how* the probability of an event is determined. In some cases, the probability of an event can be derived from counting arguments. For instance, given the roll of a fair die, we know that there only six possible events, and that all events are equally likely, so that the probability of rolling, say, a 1, is 1/6. This is called *objective* probability.

Another way of computing objective probabilities is to define the probability of an event as being the limit of a counting process, as the next example shows.

**Example 2: (Probability as a limit)**

Consider a measurement device that measures the packet header types of every packet that crosses a link. Suppose that during the course of a day the device samples 1,000,000 packets and of these 450,000 packets are UDP packets, 500,000 packets are TCP packets, and the rest are from other transport protocols. Given the large number of underlying events, to a first approximation, we can consider that the probability that a randomly selected packet uses the UDP protocol to be 450,000/1,000,000 = 0.45. More precisely, we state:

$$P(UDP) = \underset{t \to \infty}{Lim} (UDPCount(t))/(TotalPacketCoun(t))$$

where *UDPCount(t)* is the number of UDP packets seen during a measurement interval of duration *t*, and *TotalPacketCount(t)* is the total number of packets seen during the same measurement interval. Similarly *P(TCP) = 0.5*.

Note that in reality the mathematical limit cannot be achieved. Worse, over the course of a week or a month the underlying workload could change, so that the limit may not even exist. Therefore, in practice, we are forced to choose ‘sufficiently large’ packet counts and hope that the ratio thus computed correspond to a probability. This approach is also called the *frequentist* approach to probability.

[]

We can also use probabilities to characterize events for which probabilities are *subjectively* assessed.

**Example 3: (Subjective probability and its measurement)**

Consider a horse race where a favoured horse is likely to win, but this is by no means assured. We can associate a subjective probability with the event, say 0.8. Similarly, a doctor may look at a patient’s symptoms and associate them with a 0.25 probability of a particular disease. Intuitively, this measures the degree of confidence that an event will occur, based on expert knowledge of the situation that is not (or cannot be) formally stated.

How is subjective probability to be determined? A common approach is to measure the odds that a knowledgeable person would bet on that event. Continuing with the example, if a bettor really thought that the favourite would win with a probability of 0.8, then the bettor should be willing to bet $1 under the terms: if the horse wins, the bettor gets $1.25, and if the horse loses, the bettor gets $0. With this bet, the bettor expects to not lose money, and if the reward is greater than $1.25, the bettor will expect to make money. So, we can elicit the implicit subjective probability by offering a certain reward, and then lowering it until the bettor is just about to walk away, which would be at the $1.25 mark.

[]

## *1.2 Conditional probability*

Thus far, we have considered single events in isolation. We now turn our attention to sequences of events. To begin with, consider two events *E* and *F* that happen one after the other. Suppose that the probability of *E* is *P(E)* and the probability of *F* is *P(F)*. Now, suppose that we are informed that event *E* actually occurred. What can we say about *P(F)*? There are only two possibilities:

1. *P(F)* is unchanged: In other words, knowing that *E* occurred does not affect the probability of *F* occurring. In this case, *E* and *F* are said to be *independent*.
2. *P(F)* changes. If so, we denote the probability of *F*, given that *E* has occurred, by *P(F|E)* read as ‘probability of *F* given *E*’ or ‘probability of *F* conditional on *E*.’ By definition:

$$P(F|E) = \frac{P(EF)}{P(E)} \qquad \textbf{(EQ 4)}$$

It is important not to confuse *P(F|E)* and *P(EF)*. *EF* means that both *E* and *F* occurred in the original sample space. In contrast, *F|E* is the event *F* in a *reduced* sample space: the original space *S* from which all events that are not consistent with *E* having occurred have been removed. Explicitly keeping track of the reduced space can help avoid apparent contradictions such as the well-known *Monty Hall* problem (Example 5).

**Example 4: (Using conditional probability)**

Consider a device that samples packets on a link, as in Example 2. Suppose that measurements show that 20% of the UDP packets have a packet size of 52 bytes. Then, *P(packet size is 52 bytes long | packet is a UDP packet)* = 0.2. In Example 2, we computed that *P(packet is a UDP packet) = P(UDP)* = 0.45. Therefore, we can compute that *P(packet is a UDP packet <u>and</u> is 52 bytes long)* = 0.2 * 0.45 = 0.09. That is, if we were to pick a packet at random from the sample, there is a 9% chance that is a UDP packet of length 52 bytes (but it has a 20% chance of being of length 52 bytes if we know already that it is a UDP packet).

Note that to compute *P(packet is 52 bytes long)* we need to know the packet sizes for TCP and other traffic as well. For instance, if *P(packet size is 52 bytes long| packet is a TCP packet)* = 0.9 and all other packets were known to be of length 100 bytes, then *P(packet is 52 bytes long)* = 0.2 * 0.45 + 0.9 * 0.5 = 0.54.

[]

**Example 5: The Monty Hall problem**

Consider a television show (loosely modelled on a similar show hosted by Monty Hall) where three identical doors hide two goats and a luxury car. You, the contestant, can pick any door and obtain the prize behind it. We will assume that you prefer the car to the goat. If you did not have any further information, your chance of picking the winning door is clearly 1/3. Now, suppose that after you pick one of the doors, say Door 1, the host opens one of the other doors, say Door 2, and reveals a goat behind it. Should you switch your choice to Door 3 or stay with Door 1?

*Solution*

In the beginning, the sample space is {Car behind Door 1, Car behind Door 2, Car behind Door 3} with probabilities of 1/3 each. We know that the game show host will never open a door with a car behind it - only a goat. Therefore, the reduced sample space is {Car behind Door 1, Car behind Door 3}. What are the associated probabilities? Note that the *P(Car behind Door 1}* is independent of the game show host's revelation, because he will never open Door 1. Therefore, its probability in the reduced sample space continues to be 1/3. This means that *P(Car behind Door 3)* = 2/3, so it doubles your chances to switch.

One way to understand this somewhat counterintuitive result is to realize that the game show host's actions reveal private information, that is, the location of the car. The host is forced to open a door with a goat behind it. Therefore, his actions reduce the uncertainty of what lies behind Door 3.

[]

### 1.2.1 Bayes' rule

By definition of conditional probability, *P(F|E) = P(EF)/P(E)*. We can rewrite this as:

$$P(EF) = P(F|E)P(E) \tag{EQ 5}$$

That is, the probability that both *E* and *F* occur can be thought of as the product of the probabilities of two events, first, that *E* occurs, and second, that conditional on *E*, *F* occurs.

Recall that *P(F|E)* is defined in terms of the event *F* following event *E*. Now, consider the converse: *F* is known to have occurred -- what is the probability that *E* occurred? Loosely speaking, this similar to the problem: if there is fire, there will be smoke, but if there is smoke, what is the probability that there is a fire? The probability we want is *P(E|F).* We can write this, using the definition of conditional probability as:

$$P(E|F) = \frac{P(EF)}{P(F)} = \frac{P(F|E)}{P(F)}P(E) \tag{EQ 6}$$

which is *Bayes' rule.* One way of interpreting this is that it allows us to compute the degree to which some effect or *posterior F* can be attributed to some cause or *prior E*.

**Example 6: (Bayes' rule)**

Continuing with Example 4, we want to compute the following quantity: Given that a packet is 52 bytes long, what is the probability that it is a UDP packet?

*Solution*

Let 'UDP' refer to the event that a packet is of type UDP and '52' refer to the event that the packet is of length 52 bytes. From Bayes' rule,

$$P(UDP|52) = \frac{P(52|UDP)P(UDP)}{P(52)} = \frac{0.2(0.45)}{0.54} = 0.167 \tag{EQ 7}$$

[]

We can generalize Bayes' rule when a posterior can be attributed to more than one prior. Consider a posterior *F* that is due to some set of *n* priors $E_i$ such that the priors are mutually exclusive (that is, only one of them can occur). This implies that

$\sum_{i=1}^{n} P(E_i) = 1$. Then,

$$P(E_i|F) = \frac{P(FE_i)}{P(F)} = \frac{P(F|E_i)P(E_i)}{\sum_{i=1}^{n} P(F|E_i)P(E_i)} \qquad \text{(EQ 8)}$$

$$P(F) = \sum_{i=1}^{n} P(FE_i) = \sum_{i=1}^{n} P(F|E_i)P(E_i) \qquad \text{(EQ 9)}$$

The generalized Bayes' rule (also called Bayes' Theorem) allows us to compute the probability of any one of the priors $E_i$, conditional on the occurrence of the posterior *F*. This is often interpreted as follows: we have some set of mutually exclusive hypotheses $E_i$. We conduct an experiment, whose outcome is *F*. We can then use Bayes' formula to compute the revised estimate for each hypothesis.

**Example 7 (Generalized Bayes' rule)**

Continuing with Example 6, consider the following situation: we pick a packet at random from the set of sampled packets and find that its length is not 52 bytes. What is the probability that it is a UDP packet?

*Solution*

As in Example 6, let 'UDP' refer to the event that a packet is of type UDP and '52' refer to the event that the packet is of length 52 bytes. We will denote the complement of the latter event, that is, that the packet is not of length 52 bytes by '$52^c$'.

From Bayes' rule:

$$P(UDP|52^c) = \frac{P(52^c|UDP)P(UDP)}{P(52^c|UDP)P(UDP) + P(52^c|TCP)P(TCP) + P(52^c|other)P(other)} = \frac{0.8(0.45)}{0.8(0.45) + 0.1(0.5) + 1(0.05)}$$

=.36/(.36 + 0.05 + 0.05) =.36/.46 = 0.78.

Thus, if we see a packet that is *not* 52 bytes long, it is quite likely that it is a UDP packet. Intuitively, this must be true because most TCP packets are 52 bytes long, and there aren't very many non-UDP and non-TCP packets.

[]

## *1.3 Discrete and continouous random variables*

So far, we have restricted our consideration to events, which are outcomes of experiments or observations. Events occur with some probability. However, we are often interested in abstract quantities that are derived from events and observations, but are not themselves events and observations. For example, if we throw a die twice, we may want to compute the probability that the sum of the observed faces is 4. This by itself is random, and can be associated with a probability, and, moreover, depends on some underlying random events. Yet, it is neither an event nor an observation: it is a *random variable*. More formally, a random variable is a real-valued function defined over the sample space.

A random variable is discrete if the set of values it can assume (also called its *domain*) is countable and finite. These values should be *mutually exclusive* (that is, the random variable cannot simultaneously take on more than one value) and *exhaustive* (the random variable should take on at least one of the allowed values).

**Example 8: (A discrete random variable)**

Consider a random variable *P* defined as the size of an IP packet rounded up to closest kilobyte. Then, *P* assumes values from the set {1,2,3,..., 64}. This set is both mutually exclusive and exhaustive.

[]

A random variable is continuous if the values it can take on are continuous, that is, if they are a subset of the real line.

**Example 9: (A continuous random variable)**

Consider a random variable *C* defined as the time between two consecutive packet arrivals at a port of a switch. Clearly, *C* assumes values on the positive real line. By definition, these points are exclusive and exhaustive.

[]

One can argue that given the quantum nature of the world, every continuous random variable, in reality, should be replaced by a discrete variable. However, we will not pursue this line of enquiry herein.

### 1.3.1 Distributions and cumulative density functions

Consider a discrete random variable *D* that assumes values $d_i, d_2,..., d_n$. We use the notation $p(d_i)$ to refer to the probability of the event $P(D=d_i)$. The function *p(D),* which characterizes the probability that *D* will take on each value in its domain is called the *probability mass function* of *D*. It is also sometimes called the *distribution* of *D*.

Unlike a discrete random variable, which has non-zero probability of taking on any particular value, the probability that a continuous random variable *C* will take on any specific value in its domain is always 0. Nevertheless, we can define the *density* function *f(x)* of *C* as follows: *P(C takes on a value between* $x_1$ *and* $x_2$*)* $= \int_{x_1}^{x_2} f(x)dx$ Of course, we need to ensure that $\int_{-\infty}^{\infty} f(x)dx = 1$ . Alternatively, we can think of *f(x)* being implicitly defined by the statement that a uniformly randomly chosen point *x* in the domain of *C* has a probability *f(a)*of being *a*.

If the domain of a discrete random variable *D* is totally ordered (that is, for any two values $d_1$ and $d_2$ in the domain, either $d_1 > d_2$ or $d_2 > d_1$), then we can define the cumulative density function *F(D)* by: $F(d) = \sum_{i|d_i \le d} p(d_i)$ **(EQ 10)**

Similarly, the cumulative density function of a continuous random variable *C*, denoted *F(C)* is given by

$$F(x) = \int_{-\infty}^{x} f(x)dx : \qquad \textbf{(EQ 11)}$$

**Example 10: (Cumulative density functions)**

Consider a discrete random variable *D* that can take on values {1, 2, 3, 4, 5} with probabilities {0.2, 0.1, 0.2, 0.2, 0.3} respectively. The latter set is also the probability mass function of *D*. Because the domain of *D* is totally ordered, we can define the cumulative density function *F(D)* = {0.2, 0.3, 0.5, 0.7, 1.0}.

Now, consider a continuous random variable *C* defined by the density function *f(x) = 1* in the range [0,1]. The cumulative density function *F(C) = x*. We see that, though *f(0.1) = 1*, this does not mean that the value 0.1 is certain!

Note that, by definition of cumulative probability, it is necessary that it achieve a value of 1 at right extreme of the domain.

[]

### 1.3.2 Expectation of a random variable

The *expectation* or *expected value E[D]* of a discrete random variable *D* that can take on *n* values $d_i$ with probability $p(d_i)$ is given by:

$$E[D] = \sum_{i=1}^{n} d_i p(d_i) \qquad \textbf{(EQ 12)}$$

Similarly, the expectation *E[C]* of a continuous random variable *C* with density function *f(x)* is given by

$$E[C] = \int_{-\infty}^{\infty} xf(x)dx \qquad \textbf{(EQ 13)}$$

Intuitively, the expected value of a random variable is the value we expect it to take, knowing nothing else about it. For instance, if you knew the distribution of the random variable corresponding to the time it takes for you to travel from your home to work, then, on a typical day, you expect your commute time to be the expected value of this random variable.

**Example 11 (Expectation of a discrete and a continuous random variable)**

Continuing with the random variables *C* and *D* defined in Example 10, we find

*E[D]* = 1*0.2 + 2*0.1 + 3*0.2 + 4*0.2 + 5*0.3 = 0.2 + 0.2 + 0.6 + 0.8 + 1.5 = 3.3.

Note that the expected value of *D* is actually a value it cannot assume! This is true in general for all discrete random variables. One way to interpret this is that *D* will take on values ‘close’ to its expected value, in this case, 3 or 4.

Similarly,

$$E[C] = \int_{-\infty}^{\infty} xf(x)dx = \int_{0}^{1} xdx = \left.\frac{x^2}{2}\right|_0^1 = \frac{1}{2}$$

*C* is the *uniform* distribution and its expected value is the midpoint of the domain, i.e. 0.5.

[]

The expectation of a random variable gives us a reasonable idea of how it behaves. It is important to remember, however, that two random variables with the same expectation can have rather different behaviours.

### 1.3.3 Properties of expectations

Given the importance of the expectation of a random variable, we will state, without proof, some useful properties of expectations:

1. For constants *a* and *b*,

$$E[aX + b] = aE[X] + b \qquad \textbf{(EQ 14)}$$

2. For a discrete random variable *D* with probability mass function $p(d_i)$

$$E[g(D)] = \sum_{i} g(d_i)p(d_i) \ . \qquad \textbf{(EQ 15)}$$

3. For a continuous random variable *C* with density function *f(x)*,

$$E[g(C)] = \int_{-\infty}^{\infty} g(x)f(x)dx \ . \qquad \textbf{(EQ 16)}$$

4. In general, $E[g(C)]$ is not the same as $g(E[C])$, that is, a function cannot be 'taken out' of the expectation.
5. $E[X+Y] = E[X] + E[Y]$, or, more generally,

$$E\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} E[X_i] \qquad \textbf{(EQ 17)}$$

6. If $X$ and $Y$ are independent random variables, then

$$E[XY] = E[X]E[Y] \qquad \textbf{(EQ 18)}$$

These properties can be used to prove some useful identities, as the next example shows.

**Example 12: (Variance)**

The *variance* of a random variable is defined by $V(X) = E[(X\text{-}E[X])^2]$. Intuitively, it shows how 'far away' the random variable can be from its expected value. Prove that $V(X) = E[X^2] - E[X]^2$.

$V[X] = E[(X\text{-}E[X])^2] = E[X^2 - 2XE[X] + X^2] = E[X^2] - 2E[XE[X]] + E[X]^2 = E[X^2] - 2E[X]E[X] + E[X]^2 = E[X^2] - E[X]^2$.

[]

### 1.3.4 Properties of variance

The following properties of variance can be trivially shown for both discrete and continuous random variables.

1. For constant $a$,

$$V[X+a] = V[X] \qquad \textbf{(EQ 19)}$$

2. For constant $a$,

$$V[aX] = a^2V[X] \qquad \textbf{(EQ 20)}$$

3. If $X$ and $Y$ are independent random variables,

$$V[X+Y] = V[X] + V[Y] \qquad \textbf{(EQ 21)}$$

## *1.4 Standard distributions*

We now present some discrete and continuous distributions that frequently arise when studying networking problems.

### 1.4.1 Common discrete distributions: Bernoulli, Binomial, Geometric, and Poisson

**Bernoulli distribution**

A discrete random variable $X$ is called a Bernoulli random variable if it can take only two values, 0 or 1, and its probability mass function is defined as $p(0) = 1\text{-}p$ and $p(1) = p$. We can think of $X$ as representing the result of some experiment, with $X=1$ being 'success,' with probability $p$.

**Binomial distribution**

Consider a series of *n* Bernoulli experiments where the result of each experiment is *independent* of the others. We would naturally like to know the number of successes in these *n* trials. This can be represented by a discrete random variable *X* with parameters *(n,p)* and is called a *binomial* random variable. The probability mass function of a binomial random variable with parameters *(n,p)* is given by:

$$p(i) = \binom{n}{i} p^i (1-p)^{n-i} \qquad \textbf{(EQ 22)}$$

If we set $q = 1\text{-}p$, then these are just the terms of the expansion $(p+q)^n$. The expected value of a variable that is binomially distributed with parameters *(n,p)* is *np.*

**Example 13: (Binomial random variable)**

Consider a local area network with 10 stations. Assume that, at a given moment, each node can be active with probability *p*= 0.1. What is the probability that: a) one station is active, b) five stations are active, c) all 10 stations are active?

*Solution*

Assuming that the stations are independent, the number of active stations can be modelled by a binomial disturbing with parameters (10, 0.1). From the formula for *p(i)* above, we get

a) $p(1) = \binom{10}{1}(0.1^1)0.9^9 = 0.38$

b) $p(5) = \binom{10}{5}(0.1^5)0.9^5 = 1.49 \times 10^{-3}$

c) $p(10) = \binom{10}{10}(0.1^{10})0.9^0 = 1 \times 10^{-10}$

Note how the probability of one station being active is 0.38, which is actually *greater* than the probability of any single station being active. Note also how rapidly the probability of multiple active stations drops. This is what motivates spatial statistical multiplexing; the provisioning of a link with a capacity smaller than the sum of the demands of the stations.

[]

**Geometric distribution**

Consider a sequence of independent experiments, as before, each of which succeeds with probability *p*. Unlike earlier, where we wanted to count the number of successes, we want to compute the probability mass function of a random variable *X* that represents the number of trials before the first success. Such a variable is called a *geometric* random variable and has a probability mass function:

$$p(i) = (1-p)^{i-1} p \qquad \textbf{(EQ 23)}$$

The expected value of a geometrically distributed variable with parameter *p* is *1/p*.

**Example 14: (Geometric random variable)**

Consider a link that has a loss probability of 10%. Suppose that when a packet gets lost the loss is detected and the packet is retransmitted until it is correctly received. What is the probability that it would be transmitted exactly one, two, and three times?

*Solution*

Assuming that the packet transmissions are independent events, we note that the probability of success = *p* = 0.9. Therefore, *p(1)* = $0.1^0$* 0.9 = 0.9; *p(2)* = $0.1^1$* 0.9 = 0.09;  *p(3)* = $0.1^2$* 0.9 = 0.009. Note the rapid decrease in the probability of more then two transmissions, even with a fairly high packet loss rate of 10%. Indeed, the expected number of transmissions is only 1/0.9 = 1.11.

[]

**Poisson distribution**

The Poisson distribution is widely encountered in networking situations, usually to model the arrival of packets or new end-to-end connections to a switch or router. A discrete random variable $X$ with the domain {0, 1, 2, 3,...} is said to be a Poisson random variable with parameter $\lambda$ if, for some $\lambda > 0$:

$$P(X = i) = e^{-\lambda}\left(\frac{\lambda^i}{i!}\right) \qquad \textbf{(EQ 24)}$$

The Poisson distribution (which has only a single parameter $\lambda$) can be used to model a binomial distribution with two parameters ($n$ and $p$) when $n$ is 'large' and $np$ is 'small.' In this case, the Poisson variable's parameter $\lambda$ corresponds to the product of the two binomial parameters (i.e. $\lambda = n_{Binomial} * p_{Binomial}$). Recall that a binomial distribution arises naturally when we conduct independent trials. The Poisson distribution, therefore, arises when the number of such independent trials is large, and the probability of success of each trial is small. The expected value of a Poisson distributed random variable with parameter $\lambda$ is $\lambda$.

Consider an endpoint sending a packet on a link. We can model the transmission of a packet by the endpoint in a given time interval as a trial as follows: if the source sends a packet in a particular interval, we will call the trial a success, and if the source does not send a packet, we will call the trial a failure. When the load is light, the probability of success of a trial defined in this manner, which is just the packet transmission probability, is small. Therefore, as the number of endpoints grows, and if we can assume the endpoints to be independent, the sum of their loads will be well-modelled by a Poisson random variable. This is heartening, because systems subjected to a Poisson load are mathematically tractable, as we will see in our discussion of queueing theory. Unfortunately, over the last two decades, numerous measurements have shown that actual traffic can be far from Poisson. Therefore, this modelling assumption should be used with care and only as a rough approximation to reality.

**Example 15: (Poisson random variable)**

Consider a link that can receive traffic from one of 1000 independent endpoints. Suppose that each node transmits at a uniform rate of 0.001 packets/second. What is the probability that we see at least one packet on the link during an arbitrary one-second interval?

*Solution*

Given that each node transmits packets at the rate of 0.001 packets/second, the probability that a node transmits a packet in any one-second interval is $p_{Binomial}$=0.001. Thus, the Poisson parameter $\lambda$=1000*0.001 = 1. The probability that we see at least one packet on the link during any one-second interval is therefore 1- $p(0)$ = 1-$e^{-1}1^0$/1! = 1-1/$e$ = 0.64. That is, there is a 64% chance that, during an arbitrary one-second interval, we will see one or more packets on the link.

[]

It turns out that a Poisson random variable is a good approximation to a binomial random variable even if the trials are weakly dependent. Indeed, we do not even require the trials to have equal probabilities, as long as the probability of success of each individual trial is 'small.' This is another reason why the Poisson random variable is frequently used to model the behaviour of aggregates.

### 1.4.2 Common continuous distributions: Uniform, Gaussian, Exponential, and Power-Law

Recall that, unlike discrete random variables, the domain of a continuous random variable is a subset of the real line. We now consider some common continuous distributions.

**Uniform distribution**

A random variable *X* is said to be uniformly randomly distributed in the domain [*a,b*] if its density function *p(x) = 1/(b-a)* when *x* lies in [*a,b*] and is 0 otherwise. The expected value of a uniform random variable with parameters *a,b* is *(a+b)/2*.

**Gaussian or Normal distribution**

A random variable is Gaussian or normally distributed with parameters $\mu$ and $\sigma^2$ if its density is given by:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{(x-\mu)^2}{2\sigma^2}} \qquad \textbf{(EQ 25)}$$

The Gaussian distribution is the limiting case of the binomial distribution as *n* tends to infinity. That is, if we have a very large number of independent trials, such that the random variable measures the number of trials that succeed, then the random variable is Gaussian. Thus, Gaussian random variables naturally occur when we want to study the statistical properties of aggregates.

The Gaussian distribution is called 'normal' because many quantities, such as the heights of people, the slight variations in the size of a manufactured item, and the time taken to complete an activity follow the well-known 'bell-shaped' curve. When performing experiments or simulations, it is often the case that the same quantity assumes different values during different trials. For instance, if five students were each measuring the pH of a reagent, it is likely that they would get five slightly different values. In such situations, it is common to assume that these quantities, which are supposed to be the same, are in fact normally distributed about some mean. Generally speaking, if you know that a quantity is supposed to have a certain standard value, but you also know that there can be small variations in this value, then it is reasonable to assume that the quantity is a Gaussian random variable with its mean centred around the expected value.

The expected value of a Gaussian random variable with parameters $\mu$ and $\sigma^2$ is $\mu$ and its variance is $\sigma^2$. In practice, it is often convenient to work with a *standard* Gaussian distribution, that has a zero mean and a variance of 1. It is possible to convert a Gaussian random variable *X* with parameters $\mu$ and $\sigma^2$ to a Gaussian random variable *Y* with parameters 0,1 by choosing $Y = (X - \mu)/\sigma$.

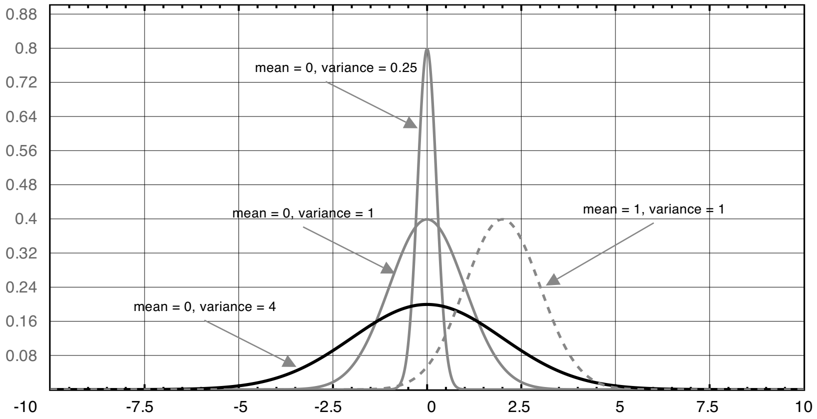


**FIGURE 1. Gaussian distribution showing different values for the mean and variance**

Here are some properties of a Gaussian variable:

1. The distribution is symmetric about the mean and asymptotes to 0 at $+\infty$ and $-\infty$. The $\sigma^2$ parameter controls the width of the central 'bell': the larger this parameter, the wider the bell, and the lower the maximum value of the density function.
2. The probability that a Gaussian random variable *X* lies between $\mu - \sigma$ and $\mu + \sigma$ is approximately 68.26%.

3. The probability that a Gaussian random variable *X* lies between $\mu - 2\sigma$ and $\mu + 2\sigma$ is approximately 95.44%.
4. The probability that a Gaussian random variable *X* lies between $\mu - 3\sigma$ and $\mu + 3\sigma$ is approximately 99.73%.
5. If *X* is Gaussian with parameters ($\mu$, $\sigma^2$), then the random variable *aX + b,* where *a* and *b* are constants, is also Gaussian, with parameters ($a\mu + b$, $(a\sigma)^2$).
6. If *X* is Gaussian with parameters ($\mu_X$, $\sigma_X^2$), and *Y* is Gaussian with parameters ($\mu_Y$, $\sigma_Y^2$), and *X* and *Y* are independent, then *X+Y* is Gaussian with parameters ($\mu_X + \mu_Y$, $\sigma_X^2 + \sigma_Y^2$), and *X-Y* is also Gaussian, with parameters ($\mu_X - \mu_Y$, $\sigma_X^2 + \sigma_Y^2$).

**Example 16: (Gaussian random variable)**

Suppose that the number of packets that arrive on a link to a router in a one-second interval can be modelled accurately by a normal distribution with parameters (20, 4). How many packets can we actually expect to see with at least 99% confidence?

*Solution*

The number of packets are distributed (20, 4), so that $\mu$ = 20 and $\sigma$ =2. From property 3 above, we have more than 99% confidence that the number of packets seen will be $\mu \pm 3\sigma$, i.e., between 14 and 26.

[]

**Exponential distribution**

A random variable *X* is exponentially distributed with parameter $\lambda$, where $\lambda > 0$ if its density function is given by:

$$f(x) = \begin{cases} \lambda e^{-\lambda x} & \text{if x >= 0} \\ 0 & \text{if x <0} \end{cases} \tag{EQ 26}$$

The expected value of such a random variable is $\frac{1}{\lambda}$ and its variance is $\frac{1}{\lambda^2}$. The exponential distribution is the continuous analogue of the geometric distribution. Recall that the geometric distribution measures the number of trials until the first success. Correspondingly, the exponential distribution arises when we are trying to measure the duration of time before some event happens (i.e. achieves success). For instance, it is used to model the time between two consecutive packet arrivals on a link.

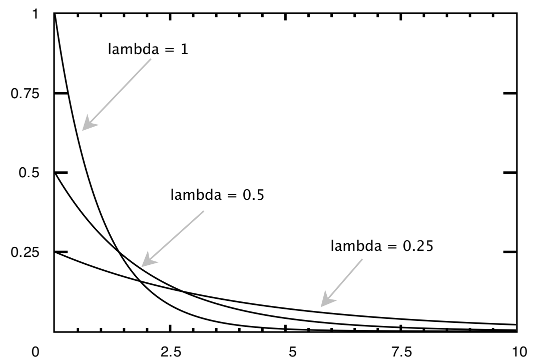


**FIGURE 2. Exponentially distributed random variables with different values of $\lambda$**

In practice, we are usually interested in the cumulative density function of the exponential distribution, *F(x)* which is given by:

$$F(x) = P(X \le x) = 1 - e^{-\lambda x} \qquad \text{(EQ 27)}$$

**Example 17: (Exponential random variable)**

Measurements show that the average length of a phone call is three minutes. Assuming that the length of a call is an exponential random variable, what is the probability that a call lasts more than six minutes?

Clearly, the $\lambda$ parameter for this distribution is 1/3. Therefore, the probability that a call lasts more than six minutes is 1-*F(6)* = $e^{-6/3}$ = $e^{-2}$ = 13.5%

[]

An important property of the exponential distribution is that it is *memoryless* and, in fact, it is the *only* memoryless distribution. Intuitively, this means that the expected remaining time until the occurrence of an event with an exponentially distributed waiting time is *independent* of the time at which the observation is made. More precisely, $P(X > s+t \mid X>s) = P(X>t)$ for all *s, t*. From a geometric perspective, if we truncate the distribution to the left of any point on the positive X axis, then rescale the remaining distribution so that the area under the curve is 1, we will obtain the original distribution. The following examples illustrate this useful property.

**Example 18: (Memorylessness 1)**

Suppose the time taken by a teller at a bank is an exponentially distributed random variable with an expected value of one minute. When you arrive at the bank, the teller is already serving a customer. If you join the queue now, you can expect to wait one minute before being served. However, suppose you decide to run an errand and return to the bank. If the same customer is still being served (i.e. the condition $X>s$), if you join the queue now, the expected waiting time for you to be served would *still* be 1 minute!

[]

**Example 19: (Memorylessness 2)**

Suppose that a switch has two parallel links to another switch and packets can be routed on either link. Consider a packet *A* that arrives when both links are already in service. Therefore, the packet will be sent on the first link that becomes free. Suppose this is link 1. Now, assuming that link service times are exponentially distributed, which packet is likely to finish transmission first: packet *A* on link 1 or the packet continuing service on link 2?

*Solution*

Because of the memorylessness of the exponential distribution, the expected remaining service time on link 2 at the time that *A* starts transmission on link 1 is exactly the same as the expected service time for *A,* so we expect both to finish transmission at the same time. Of course, we are assuming we don't know the service time for *A*. If a packet's service time is proportional to its length, and we know *A'*s length, then we no longer have an expectation for its service time: we know it precisely, and this equality no longer holds.

[]

**Power law distribution**

A random variable parametrized by its minimum value $x_{min}$ and $\alpha > 1$ is said to obey the power law distribution if its density function is given by:

$$f(x) = \frac{(\alpha - 1)}{x_{min}}\left(\frac{x}{x_{min}}\right)^{-\alpha} \qquad \text{(EQ 28)}$$

Clearly, as $x$ increases, its probability decreases rapidly (though not as rapidly as with an exponential distribution - which is why a power-law distribution is also called ‘heavy-tailed’).

Intuitively, if we have objects distributed according to a power law, then there are a few ‘elephants’ and many ‘mice’. The elephants are few and are responsible for most of the probability mass. From an engineering perspective, whenever we see such a distribution, it makes sense to build a system that deals well with the elephants, even at the expense of ignoring the mice. Two engineering rules of thumb that also follow from a power law are the *90/10 rule* (90% of the task can be accomplished in 10% of the time: the remaining 10% takes 90% of the time), and the dictum ‘*optimize for the common case*.’

Note that when $\alpha < 2$, the expected value of the random variable is infinite. In practice, this means that a system described by such a random variable is unstable (i.e. its value is unbounded). On the other hand when $\alpha > 2$, the tail probabilities fall rapidly enough that a power-law random variable can be approximated by an exponential random variable.

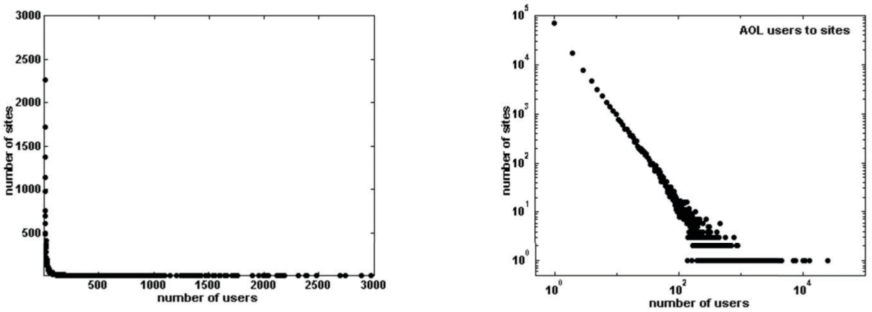


**FIGURE 3. Power law distribution: number of users visiting a web site (from [Adamic 2002])**

**Example 20: (Power law distribution)**

A widely-studied example of power-law distribution is the random variable that describes the number of users who visit one of a collection of web sites on the Internet [L. Adamic, “Zipf, Power-laws, and Pareto - a ranking tutorial, http://www.hpl.hp.com/research/idl/papers/ranking/ranking.html 2002]. An example is shown in Figure 3. On the left hand side, the plot is on a linear scale (and zooms into the lower left quadrant), and we see that thousands of sites get only one or zero visitors. A few sites get more than 2000 visitors, and none get more than 3000 visitors. The same data, when plotted on a log-log scale on the right hand side shows a linear relationship, a classic ‘signature’ of a power-law distribution.

[]

## *1.5 Useful theorems*

We now state some fundamental theorems of probability. Markov’s and Chebyshev’s inequality allow us to bound the amount of mass in a distribution in the tail knowing nothing more than the expected value. The law of large numbers allows us to relate real-world measurements with the expectation of a random variable. Finally, the central limit theorem shows why so many real-world random variables are normally distributed.

### 1.5.1 Markov’s inequality

If $X$ is a non-negative random variable with mean $\mu$, then for any constant $a > 0$

$$P(X \geq a) \leq \frac{\mu}{a} \qquad \textbf{(EQ 29)}$$

So, the expected value of $X$. Markov's inequality requires knowledge only of the mean of the distribution, so it is widely applicable. (Note that this inequality is trivial if $a < \mu$).

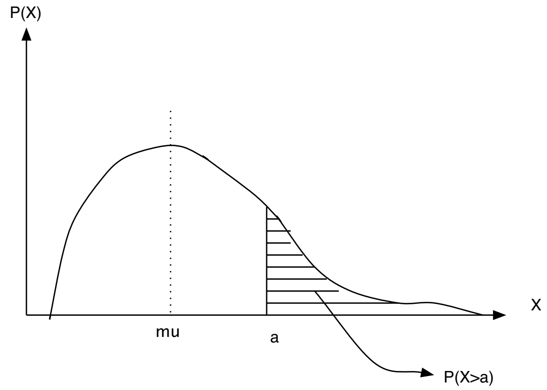


FIGURE 4. **Markov's inequality**

### 1.5.2 Chebychev's inequality

If $X$ is a random variable with a finite mean $\mu$ and variance $\sigma^2$, then for any constant $a > 0$

$$P(|X-\mu| \geq a) \leq \frac{\sigma^2}{a^2} \quad \text{(EQ 30)}$$

Chebychev's inequality bounds the 'tails' of a distribution on both sides of the mean, given the variance. Roughly, the further away we get from the mean (the larger $a$ is), the less mass there is in the tail (because the right hand size decreases by a factor quadratic in $a$).

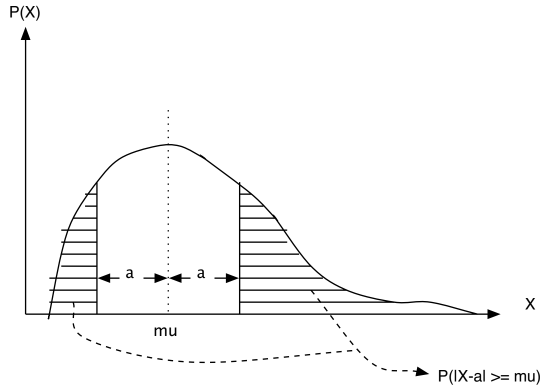


FIGURE 5. **Chebychev's inequality**

### 1.5.3 Strong law of large numbers

The law of large numbers relates the *sample mean*--the average of a set of observations of a random variable--with the *population* or *true* mean, which is its actual mean (expected value). The *strong* law of large numbers, the better-known variant, states that if $X_1, X_2,..., X_n$ are $n$ independent, identically distributed random variables with the same expected value $\mu$, then:

$$P\left(\lim_{n \to \infty}(X_1 + X_2 + \ldots + X_n)/n = \mu\right) = 1 \quad \text{(EQ 31)}$$

No matter how *X* is distributed, if we compute the average of a large number of observations, we will surely find the true mean. This is the basis of a variety of statistical techniques for hypothesis testing, as described in the chapter on statistics.

### 1.5.4 Central limit theorem

The central limit theorem deals with the sum of a *large* number of *independent* random variables that are arbitrarily distributed. No matter how each random variable is distributed, as long as its contribution to the total is 'small,' the sum is well described by a Gaussian random variable. More precisely, let $X_1, X_2,..., X_n$ be *n* independent, identically distributed random variables, each with a finite mean $\mu$ and variance $\sigma^2$. Then, the distribution of the normalized sum given by $\frac{X_1 + \ldots + X_n - n\mu}{\sigma\sqrt{n}}$ tends to the standard (0,1) normal as $n \to \infty$. The central limit theorem states more precisely why the Gaussian is the limit of the binomial distribution.

In practice, the central limit theorem allows us to model aggregates by a Gaussian random variable if the size of the aggregate is large and the elements of the aggregate are independent.

## *1.6 Jointly distributed random variables*

So far, we have considered distributions of one random variable. We now consider situations where we want to study the distribution of two random variables simultaneously.

**Example 20: (Joint probability distribution)**

Consider the two events: 'rain today' and 'rain tomorrow.' Let the random variable *X* be 0 if it does not rain today and 1 if it does. Similarly, let the random variable *Y* be 0 if it does not rain tomorrow and 1 if it does. There are four possible values for the random variables *X* and *Y* considered together: 00, 01, 10, and 11, corresponding to four joint events. We can associate probabilities with these events with the usual restrictions that these probabilities lie in [0,1] and that their sum be 1. For instance, consider the following distribution:

*P(00)* = 0.2;

*P(01)* = 0.4;

*P(10)* = 0.3;

*P(11)* = 0.1.

This is the *joint probability* distribution of *X* and *Y*. Given this joint distribution, we can extract the distribution of *X* alone, which is simply the probabilities for *X=0* and *X=1.* Clearly, we can obtain *P(X=0)* as *P(00) + P(01)* = 0.1 + 0.4 = 0.6. Similarly, *P(X=1)* = 0.3 + 0.1 = 0.4. As expected *P(X=0) + P(X=1)* = 1. Similarly, note that *P(Y=0)* = 0.5 and *P(Y=1)* = 0.5.

[]

We call the distribution of *X* alone as the *marginal* distribution of *X* and denote it $p_X$. Similarly, the marginal distribution of *Y* is denoted $p_Y$. Generalizing from the example above, we see that to obtain the marginal distribution of *X*, we should set *X* to each value in its domain and then sum over *all possible values of Y*. Similarly, to obtain the marginal distribution of *Y*, we set *Y* to each value in its domain and sum over all possible values of *X*.

An important special case of a joint distribution is when the two variables *X* and *Y* are *independent*. Then, $p_{XY}(xy)$ *=P(X=x AND Y=y) = P(X=x) * P(Y=y)=*$p_X(x)p_Y(y)$. That is, each entry in the joint distribution is obtained simply as the product of the marginal distributions corresponding to that value. We can also state this as $p(x,y) = p_X(x)p_Y(y)$.

**Example 21: (Independence)**

In Example 20, $p_{XY}(00)$ = 0.2, $p_X(0)$ = 0.6 and $p_Y(0)$ = 0.5, so *X* and *Y* are not independent, and we cannot decompose the joint distribution into the product of the marginal distributions.

[]

Given the joint distribution, we can compute the *conditional probability mass function of X*, denoted by $p_{X|Y}(x|y)$ by $P(X=x \mid Y=y) = P(X=x \text{ AND } Y=y)/P(Y=y) = p_{XY}(xy)/p_Y(y)$.

**Example 22: (Conditional probability mass function)**

Continuing with Example 21, suppose we wanted to compute the probability that it will rain tomorrow, given that it rained today. This is $p_{Y|X}(1|1) = p_{XY}(11)/p_X(1)$ = 0.1/0.4 = 0.25. Thus, knowing that it rained today makes it more probable that it will rain tomorrow.

[]

We can generalize the notion of joint probability in several ways. We outline these generalizations next. Note that the concepts we have developed for the simple case above continue to hold for these generalizations.

1. Instead of having only two values, 0 and 1, *X* and *Y* could assume any number of finite discrete values. In this case, if there are *n* values of *X* and *m* values of *Y*, we would need to specify, for the joint distribution, a total of *nm* values. If *X* and *Y* are independent, however, we only need to specify *n+m* values to specify the joint distribution.
2. We can generalize this further and allow *X* and *Y* to be continuous random variables. Then, the joint probability distribution $p_{XY}(xy)$ is implicitly defined by:

$$P(a < X < a+\alpha, b < Y < b+\beta) = \int_{b}^{(b+\beta)} \int_{a}^{(a+\alpha)} p_{XY}(xy)\,dx\,dy \qquad \textbf{(EQ 32)}$$

   Intuitively, this is the probability that a randomly chosen two-dimensional vector will be in the vicinity of *(a,b)*.
3. As a further generalization, consider the joint distribution of *n* random variables, $X_1, X_2,..., X_n$, where each variable is either discrete or continuous. If they are all discrete, then we need to define the probability of each possible choice of each value of $X_i$. This grows exponentially with the number of random variables and with the size of each domain of each variable. Thus, it is impractical to completely specify the joint probability distribution for a large number of variables. Instead, we exploit pairwise independence between the variables, using the construct of a Bayesian network, which is described next.

### 1.6.1 Bayesian networks

Bayes' rule allows us to compute the degree to which one of a set of mutually exclusive prior events contribute to a posterior condition. Suppose the posterior condition was itself a prior to yet another posterior and so on. We could then imagine tracing this chain of conditional causation back from the final condition to the initial causes. This, in essence, is a Bayesian network.

More formally, a Bayesian network is a directed acyclic graph whose vertices represent random variables and whose edges represent conditional causation between these events: there is an edge from an event $E_i$, called the 'parent' or 'cause', to every $E_j$ whose outcome depends on it, called its 'children' or 'effects.' If there is no edge between $E_i$ and $E_j$, they are independent events. Each node also stores the conditional probability distribution $P(E_i|\, parents(E_i))$. The network allows us to compute $P(\mathbf{E}) = P(E_1E_2E_3...E_n)$ for all values of $E_i$, their joint distribution as:

$$P(\mathbf{E}) = \prod_{i=1}^{n} P(E_i|parents(E_i)) \qquad \textbf{(EQ 33)}$$

That is, the joint distribution is simply the product of the *local distributions*, which are stored at each node. This greatly reduces the amount of information required to describe the joint probability distribution of the random variables.

Choosing the Bayesian graph is a non-trivial problem and one that we will not discuss further. An overview can be found in [D. Heckerman, "A Tutorial on Learning with Bayesian Networks," Microsoft Research Technical Report 95-06, March 1995 and Russell and Norvig].

Note that, because the Bayesian network encodes the joint distribution for ***E***, in principle, we can extract any probability we want from it. Usually we want to compute something much simpler. A Bayesian network allows us to compute probabilities of interest without having to compute the entire joint distribution, as the next example demonstrates.

**Example 23: (Bayesian network)**

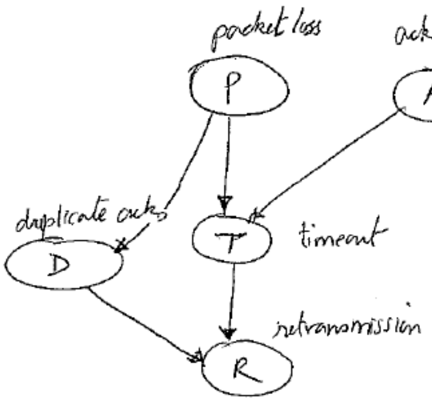


FIGURE 6. **A Bayesian network to represent TCP retransmissions**

Consider the Bayesian network in Figure 6. It shows that when there is a packet loss (cause) there can be a timeout at TCP transmitter (effect). Similarly, on the loss of an acknowledgement, there can also be a timeout. A packet loss can lead to a duplicate acknowledgment being received at the transmitter. Packet and ack loss are mutually exclusive, as are duplicate acks and timeouts. And, finally, if there is either a duplicate ack or a timeout at the transmitter, it will retransmit a packet. For notational convenience, we represent the events packet loss, ack loss, duplicate ack, timeout, and retransmission by *P, A, D, T,* and *R* respectively.

The joint distribution of the events (*P, A, D, T, R*) would assign a probability to every possible combination of the events, such as *P(packet loss AND no ack loss AND no duplicate ack AND timeout AND no retransmission)*. In practice, we rarely need the joint distribution. Instead, we may only be interested in computing the following probability: *P(packet loss | retransmission) = P(P|R)*. That is, we observe that the transmitter has retransmitted a packet. What is the probability of packet loss, i.e. what is *P(P|R)*?

For notational simplicity, let $r = P(R)$, $p=P(P)$, $t = P(T)$, $a = P(A)$ and $d=P(D)$. From the network, it is clear that we can write $P(R)$ as $P(R|T)t + P(R|D)d$. Similarly, $P(T) = t = P(T|P)p + P(T|A)a$ and $P(D) = d = P(D|P)p$. Therefore,

$$P(R) = P(R|T)(P(T|P)p + P(T|A)a) + P(R|D)P(D|P)p.$$

If we know *a* and *p* and the conditional probabilities corresponding to each link, we can therefore compute *r*.

From the definition of conditional probabilities, *P(P|R) = P(P AND R)/r.* We have already seen how to compute the denominator. To compute the numerator, we sum across all possibilities for *P* and *R* as follows:

*P(P AND R) = P(P AND R AND T AND D) + P(P AND R AND T AND NOT D) + P(P AND R AND NOT T AND D) + P(P AND R AND NOT T AND NOT D).*

However, note that *T* and *D* are mutually exclusive, so

*P(T AND D) = 0* and *P(T AND NOT D) = P(T)* and

*P(NOT T AND D) = P(D)*.

Thus,

*P(P AND R) = P(P AND R AND T) + P(P AND R AND D) + P(P AND R AND NOT T AND NOT D)*.

Note also the last term is 0, because we do not have a retransmission unless there is either a timeout or a duplicate ack. Thus,

*P(P AND R) = P(P AND R AND T) + P(P AND R AND D)*.

Replacing this in the expression for conditional probability, we get:

$$P(P|R) = \frac{P(P \wedge R \wedge T) + P(P \wedge R \wedge D)}{P(R|T)(P(T|P)p + P(T|A)a) + P(R|D)P(D|P)p}$$

All these variables can be computed by straightforward observations. For instance, to compute $P(P \wedge R \wedge T)$, we can compute the ratio of all retransmissions where there was both a packet loss and timeout event to the number of transmissions. Similarly, to compute *P(R|T)*, we can compute the ratio of the number of times a retransmission happens due to a timeout to the number of times a timeout happens.

[]

## 1.7 Exercises

1 **Sample space**

In the IEEE 802.11 protocol, the congestion window (CW) parameter is used as follows: initially, a terminal waits for a random time period (called *backoff*) chosen in the range [1, $2^{CW}$] before sending a packet. If an acknowledgement for the packet is not received in time, then CW is doubled, and the process is repeated, until CW reaches the value CWMAX. The intial value of CW is CWMIN. What is the sample space for (a) the value of CW? (b) the value of the the backoff?

2 **Interpretations of probability**

Consider the statement: given the conditions right now, the probability of a snowstorm tomorrow morning is 25%. How would you interpret this statement from the perspective of an objective, frequentist, and subjective interpretation of probability (assuming these are possible)?

3 **Conditional probability**

Consider a device that samples packets on a link. (a) Suppose that measurements show that 20% of packets are UDP, and that 10% of all packets are UDP packets with a packet size of 100 bytes.What is the conditional probability that a UDP packet has size 100 bytes? (b) Suppose 50% of packets were UDP, and 50% of UDP packets were 100 bytes long. What fraction of all packets are 100 byte UDP packets?

4 **Conditional probability again**

Continuing with Ex. 3: How does the knowledge of the protocol type change the sample space of possible packet lengths? In other words, what is the sample space before and after you know the protocol type of a packet?

5 **Bayes' rule**

For Exercise 3(a), what additional information do you need to compute P(UDP|100)? Setting that value to *x*, express P(UDP|100) in terms of *x*.

6 **Cumulative distribution function**

(a) Suppose discrete random variable *D* take values {1, 2, 3, ...,i,...} with probability $1/2^i$. What is its CDF?

(b) Suppose continuous random variable $C$ is uniform in the range $[x_1, x_2]$. What is its CDF?

### 7 Expectations

Compute the expectations of the $D$ and $C$ in Exercise 6.

### 8 Variance

Prove that $V[aX] = a^2 V[X]$.

### 9 Bernoulli distribution

A hotel has 20 guest rooms. Assuming outgoing calls are independent and that a guest room makes 10 minutes worth of outgoing calls during the busiest hour of the day, what is the probability that 5 calls are simultaneously active during the busiest hour? What is the probability of 15 simultaneous calls?

### 10 Geometric distribution

Consider a link that has a packet loss rate of 10%. Suppose that every packet transmission has to be acknowledged. Compute the expected number of data transmissions for a successful packet+ack transfer.

### 11 Poisson distribution

Consider a binomially distributed random variable $X$ with parameters $n$=10, $p$=0.1. (a) Compute the value of $P(X=8)$ using both the binomial distribution and the Poisson approximation. (b) Repeat for $n$=100, $p$=0.1

### 12 Gaussian distribution

Prove that if $X$ is Gaussian with parameters $(\mu, \sigma^2)$, then the random variable $Y=aX + b$, where $a$ and $b$ are constants, is also Gaussian, with parameters$(a\mu + b, (a\sigma)^2)$.

### 13 Exponential distribution

Suppose that customers arrive to a bank with an exponentially distributed inter-arrival time with mean 5 minutes. A customer walks into the bank at 3pm. What is the probability that the next customer arrives no sooner than 3:15?

### 14 Exponential distribution

It is late August and you are watching the Perseid meteor shower. You are told that that the time between meteors is exponentially distributed with a mean of 200 seconds. At 10:05 pm, you see a meteor, after which you head to the kitchen for a bowl of icecream, returning outside at 10:08pm. How long do you expect to wait to see the next meteor?

### 15 Power law

Consider a power-law distribution with $x_{min}$ = 1 and $\alpha = 2$ and an exponential distribution with $\lambda$ = 2. Fill in the following table:

| x | $f_{power\_law}(x)$ | $f_{exponential}(x)$ |
|---|---|---|
| 1 | | |
| 5 | | |
| 10 | | |
| 50 | | |
| 100 | | |

It should now be obvious why a power-law distribution is called ‘heavy-tailed’!

### 16 Markov’s inequality

Consider a random variable $X$ that exponentially distributed with parameter $\lambda$ = 2. What is the probability that $X >$ 10 using (a) the exponential distribution (b) Markov’s inequality.

### 17 Joint probability distribution

Consider the following probability mass function defined jointly over the random variables, $X$, $Y$, and $Z$:

P(000) = 0.05; P(001) = 0.05; P(010) = 0.1; P(011)=0.3;P(100) = 0.05; P(101) = 0.05; P(110) = 0.1; P(111)=0.3. (a) Write down $p_X$, $p_Y$,$p_Z$,$p_{XY}$,$p_{XZ}$,$p_{YZ}$. (b) Are X and Y, X and Z, or Y and Z independent? What is the probability that X=0 given that Z=1?

# CHAPTER 2 *A Review of Statistics*

This chapter reviews basic statistical concepts and techniques. We start with an overview of the moments of a distribution and a useful tool called the moment generating function that we use to study the normal distribution in greater depth. We then discuss statistical techniques to deal with some situations that arise frequently in carrying out research in computer networking: describing data parsimoniously, inferring the parameters of a population from a sample, comparing outcomes, and inferring or independence of variables. We conclude with some approaches to dealing with large data sets and a description of common mistakes in statistical analysis (and how to avoid them).

## *2.1 Background*

In this section we review moments about zero and the mean, and the description of moments using a moment generating function (abbreviated MGF). We also study the normal distribution using the MGF, which allows us to concisely state and prove the Central Limit Theorem.

### 2.1.1 Moments

Recall that the expected value of a random variable *X* (also called its mean μ) is defined by:

$$E(X) = \sum_x xP(x) \quad \text{for a discrete variable with probability mass function P(x)}$$
$$E(X) = \int xf(x)dx \qquad \text{for a continous variable with density function f(x)} \qquad \textbf{(EQ 1)}$$

Also, by definition, if $Y = g(X)$ is some function of *X*, such as $X^2$ or $e^X$, then:

$$E(Y) = E(g(x)) = \sum_x g(x)P(x) \qquad \text{for a discrete variable}$$
$$E(Y) = E(g(x)) = \int g(x)f(x)dx \qquad \text{for a continous variable} \qquad \textbf{(EQ 2)}$$

**Example 1: Expected value of a function of a variable**

Let *X* be a uniform random variable defined in the interval [0,1]. Then, $E(X^2) = \int_0^1 x^2 dx = \frac{1}{3}x^3 \Big|_0^1 = \frac{1}{3}$.

[]

The moments of a distribution are a set of parameters that summarize it. Given a random variable *X*, its first moment about the origin is its mean $E(X) = \mu$. Its second moment about the origin is the expected value of the random variable $X^2$, i.e., $E(X^2)$. The $r^{\text{th}}$ moment of *X* about the origin, denoted $\mu_r{}'$, is defined as $E(X^r)$. Note that $\mu_1{}'=\mu$.

We also define the $r^{th}$ moment about the mean, denoted $\mu_r$, by $E((X-\mu)^r)$. Note that the variance of the distribution, denoted by $\sigma^2= \mu_2$. The third moment about the mean $\mu_3$ is used to construct a measure of *skewness* (which describes whether the probability mass is more to the left or the right of the mean, compared to a normal distribution) and the fourth moment about the mean $\mu_4$ is used to construct a measure of peakedness or *kurtosis*, which measures the width of a distribution compared to a normal distribution.

The two definitions of a moment are related. For example, it is easy to show that the variance of *X*, denoted *V(X)*, can be computed as $V(X) = E(X^2) - (E(X))^2$. Therefore, $\mu_2 = \mu_2{}' - \mu^2$. Similar relationships can be found between the higher moments by writing out the terms of the binomial expansion of $(X-\mu)^r$.

### 2.1.2 Moment generating functions

Except under some pathological conditions, a distribution can be thought to be uniquely represented by its moments. That is, if two distributions have the same moments, then, except under some rather unusual circumstances, they will be identical. Therefore, it is convenient to have an expression (or 'fingerprint') that compactly represents all the moments of a distribution. Such an expression should have terms corresponding to $\mu_r{}'$ for all values of *r*.

We can get a hint regarding a suitable representation from the expansion of $e^x$:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots \qquad \textbf{(EQ 3)}$$

We see that there is one term for each power of *x*. This motivates the definition of the moment generating function (MGF) of a random variable *X* as the expected value of $e^{tX}$, where *t* is an auxiliary variable:

$$M(t) = E(e^{tX}) \qquad \textbf{(EQ 4)}$$

To see how this represents the moments of a distribution, we expand *M(t)* as

$$\begin{aligned} M(t) = E(e^{tX}) &= E\left(1 + (tX) + \left(\frac{t^2X^2}{2!}\right) + \left(\frac{t^3X^3}{3!}\right) + \ldots\right) \\ &= 1 + E(tX) + E\left(\frac{t^2X^2}{2!}\right) + E\left(\frac{t^3X^3}{3!}\right) + \ldots \\ &= 1 + tE(X) + \frac{t^2}{2!}E(X^2) + \frac{t^3}{3!}E(t^3X^3) + \ldots \\ &= 1 + t\mu_1' + \frac{t^2}{2!}\mu_2' + \frac{t^3}{3!}\mu_3' + \ldots \end{aligned} \qquad \textbf{(EQ 5)}$$

Thus, the MGF represents all the moments of the random variable *X* in a single compact expression. Note that the MGF of a distribution is undefined if one or more of its moments are infinite.

We can extract all the moments of the distribution from the MGF as follows: if we differentiate *M(t)* once, the only term that is not multiplied by *t* or a power of *t* is $\mu_1{}'$. So, if we were to set *t* = 0, the value of *dM(t)/dt* at *t=0* would be $\mu_1{}'$. Similarly, if we were to differentiate *M(t)* twice, and set *t* to 0, the value of $d^2M(t)/dt^2 = \mu_2{}'$. Generalizing, it is easy to show that to get the $r^{th}$ moment of a random variable *X* about the origin, we only need to differentiate its MGF *r* times with respect to *t* and then set *t* to 0.

It is important to remember that the 'true' form of the MGF is the series expansion in Equation 5. The exponential is merely a convenient representation that has the property that operations on the series (as a whole) result in corresponding operations being carried out in the compact form. For example, it can be shown that the series resulting from the product of $e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots$ and $e^y = 1 + y + \frac{y^2}{2!} + \frac{y^3}{3!} + \ldots$ is $1 + (x+y) + \frac{(x+y)^2}{2!} + \frac{(x+y)^3}{3!} + \ldots = e^{x+y}$. This simplifies the computation of operations on the series. However, it is sometimes necessary to revert to the series representation for certain operations. In particular, if the compact notation of *M(t)* is not differentiable at *t* = 0, then we must revert to the series to evaluate *M(0)*, as shown next.

**Example 2: MGF of the standard uniform distribution**

Let *X* be a standard uniform random variable (i.e., defined in the interval [0,1]). We would like to find all its moments. We find that $M(t) = E(e^{tX}) = \int_0^1 e^{tx}dx = \frac{1}{t}e^{tx}\Big|_0^1 = \frac{1}{t}[e^t - 1]$. However, this function is not defined--and therefore not differentiable--at *t* = 0. Instead, we revert to the series:

$$\frac{1}{t}[e^t - 1] = \frac{1}{t}\left[t + \frac{t^2}{2!} + \frac{t^3}{3!} + \ldots\right] = 1 + \frac{t}{2!} + \frac{t^2}{3!} + \ldots \qquad \textbf{(EQ 6)}$$

which *is* differentiable term by term. Differentiating *r* times and setting *t* to 0, we find that $\mu_r{}' = 1/(r+1)$. So, $\mu_1{}' = \mu = 1/(1+1) = 1/2$ is the mean, and $\mu_2{}' = 1/(1+2) = 1/3 = E(X^2)$ as we found in Example 1. Note that we found the expression for *M(t)* using the compact notation, but reverted to the series for differentiating it. The justification is that the integral of the compact form is identical to the summation of the integrals of the individual terms (also see Exercise 2).

[]

Note that the MGF of some random variables is undefined because one or more of their moments are infinite.

### 2.1.3 Properties of moment generating functions

We now prove some useful properties of MGFs.

(a) If *X* and *Y* are two independent random variables, the MGF of their sum is the product of their MGFs. For, if their individual MGFs are $M_1(t)$ and $M_2(t)$ respectively, the MGF of their sum is:

$$M(t) = E(e^{t(X+Y)}) = E(e^{tX}e^{tY}) = E(e^{tX})E(e^{tY}) \text{ (from independence) } = M_1(t).M_2(t)$$

**Example 3: MGF of the sum**

Find the MGF of the sum of two independent [0,1] uniform random variables.

From Example 2, the MGF of a standard uniform random variable is $\frac{1}{t}[e^t - 1]$, so the MGF of random variable *X* defined as the sum of two independent uniform variables is $\frac{1}{t^2}[e^t - 1]^2$.

[]

(b) If random variable $X$ has MGF $M(t)$ then the MGF of random variable $Y = a+bX$ is $e^{at}M(bt)$. This is because:

$$E(e^{tY}) = E(e^{t(a+bX)}) = E(e^{at}e^{bXt}) = e^{at}E(e^{btX}) = e^{at}M(bt)$$

As a corollary, if $M(t)$ is the MGF of a random variable $X$, then the MGF of $(X-\mu)$ is given by $e^{-\mu t}M(t)$. The moments about the origin of $(X-\mu)$ are the moments about the mean of $X$. So, to compute the $r^{\text{th}}$ moment about the mean for a random variable $X$, we can differentiate $e^{-\mu t}M(t)$ $r$ times with respect to $t$ and set $t$ to 0.

**Example 4: Variance of a standard uniform random variable**

The MGF of a standard uniform random variable $X$ is $\frac{1}{t}[e^t - 1]$, so, the MGF of $(X-\mu)$ is given by $\frac{e^{-\mu t}}{t}[e^t - 1]$. To find the variance of a standard uniform random variable, we need to differentiate twice with respect to $t$ and then set $t$ to 0. Given the $t$ in the denominator, it is convenient to rewrite the expression as $\left(1 - \mu t + \frac{\mu^2 t^2}{2!} + \ldots\right)\left(1 + \frac{t}{2!} + \frac{t^2}{3!} + \ldots\right)$, where the ellipses refer to terms with third and higher powers of $t$, which will reduce to 0 when $t$ is set to 0. In this product, we need only consider the coefficient of $t^2$ (why?), which is $\frac{1}{3!} - \frac{\mu}{2!} + \frac{\mu^2}{2!}$. Differentiating the expression twice results in multiplying the coefficient by 2, and when we set $t$ to zero, we obtain $E((X-\mu)^2) = V(X) = 1/12$.

[]

These two theorems allow us to compute the MGF of a complex random variable that can be decomposed into the linear combination of simpler variables. In particular, it allows us to compute the MGF of independent, identically distributed (i.i.d) random variables, a situation that arises frequently in practice.

### 2.1.4 The normal distribution revisited

A continuous random variable $X$ is normally distributed with mean $\mu$ and variance $\sigma^2$ (denoted $X \sim N(\mu,\sigma^2)$) if the density function of $X$, denoted $f(x)$ is given by

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}}e^{\frac{-(x-\mu)^2}{2\sigma^2}} \qquad -\infty \le x \le \infty \tag{EQ 7}$$

The MGF of the normal distribution is given by

$$\begin{aligned} M(t) &= \frac{1}{\sigma\sqrt{2\pi}}\int_{-\infty}^{\infty} e^{tx - \frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}dx \\ &= \frac{e^{\mu t + \frac{1}{2}\sigma^2 t^2}}{\sigma\sqrt{2\pi}}\int_{-\infty}^{\infty} e^{-\frac{1}{2}\frac{(x-\mu-\sigma^2 t)^2}{\sigma^2}}dx \\ &= e^{\mu t + \frac{1}{2}\sigma^2 t^2} \end{aligned} \tag{EQ 8}$$

where in the last step, we recognize that the integral is the area under a normal curve, which evaluates to $\frac{1}{\sigma\sqrt{2\pi}}$. Note that the MGF of a standard normal variable with zero mean and a variance of 1 is therefore

$$M(t) = e^{\frac{1}{2}t^2} \tag{EQ 9}$$

We can use the MGF of a normal distribution to prove some elementary facts about it:

(a) If $X \sim N(\mu,\sigma^2)$ then $a + bX \sim N(a+b\mu, b^2\sigma^2)$. This is because the MGF of $a+bX$ is $e^{at}M(bt) = e^{at}e^{\mu bt + \frac{1}{2}\sigma^2(bt)^2}$ $= e^{(a+\mu b)t + \frac{1}{2}(\sigma^2 b^2)t^2}$, which can be seen to be a normally distributed random variable with mean $a + b\mu$ and variance $b^2\sigma^2$.

(b) If $X \sim N(\mu,\sigma^2)$ then $Z = (X-\mu)/\sigma \sim N(0,1)$. This is obtained trivially by substituting for $a$ and $b$ in the expression above. $Z$ is called the *standard normal variable*.

(c) If $X \sim N(\mu_1,\sigma_1^2)$ and $Y \sim N(\mu_2,\sigma_2^2)$ and $X$ and $Y$ are independent, then $X+Y \sim N(\mu_{1+}\mu_2, \sigma_1^2+\sigma_2^2)$. This is because the MGF of their sum is the product of their individual MGFs $= e^{\mu_1 t + \frac{1}{2}\sigma_1^2 t^2} e^{\mu_2 t + \frac{1}{2}\sigma_2^2 t^2} = e^{(\mu_1+\mu_2)t + \frac{1}{2}(\sigma_1^2+\sigma_2^2)t^2}$. As a corollary, the sum of any number of normal variables is also normally distributed with the mean as the sum of the individual means and the variance as the sum of the individual variances.

### 2.1.5 The Central limit theorem

The normal distribution plays a central role in statistics because of the central limit theorem, which states that the sum of a large number of *independent* random variables, no matter what distribution each individual follows, is approximately normally distributed. Consider a set of measurements of a physical system. Each measurement can be modelled as an independent random variable whose mean and variance are those of the population. From the central limit theorem, their sum, and therefore their mean (which is just the normalized sum) is approximately normally distributed. As we will show below, this allows us to infer the true (or *population*) mean from the mean of the measurements (the *sample* mean), which forms the foundation of statistical confidence. We now prove this theorem using MGFs [This proof closely follows Bulmer *Principles of Statistics*, Dover 1979 pp. 115].

The proof proceeds in three stages. First, we compute the MGF of the sum of $n$ random variables in terms of the MGFs of each of the random variables. Second, we find a simple expression for the MGF of a random variable when the variance is large (a situation we expect when adding together many independent random variables). Finally, we plug in this simple expression back into the MGF of the sum to obtain the desired result. These stages are explicitly marked in the proof below.

*Preliminaries and definitions*: Consider a random variable $Y = X^1 + X^2 ... + X^n$, the sum of $n$ independent random variables. We denote the $r$th moment about the mean of the $i$th random variable by $\mu_r^i$ and the corresponding moment about the mean of $Y$ by $\mu_r$. Similarly, we denote the standard deviation of each $X^i$ by $\sigma^i$ and the standard deviation of $Y$ by $\sigma$. Because the $X^i$s are independent,

$$\mu = \sum \mu^i \; ; \; \sigma^2 = \sum (\sigma^i)^2 \qquad \textbf{(EQ 10)}$$

We denote the MGF of the $i$th random variable $X^i$ by $M^i(t)$ and the MGF of the variable $W^i = (X^i - \mu_i)$ by $N^i(t)$. By definition of $W^i$, the $r$th moment of $W^i$ about the origin is the $r$th moment of $X^i$ about its mean. Also, because the $X^i$ are independent, so are the $W^i$.

*Stage 1:* Now, $Y - \mu = X^1 + X^2 ... + X - \sum \mu^i = \sum (X^i - \mu^i) = \sum W^i$. So the MGF of $(Y - \mu)$ is the product of the MGFs of the $W^i$= $\prod_{i=1}^{n} N^i(t)$ and the MGF of $(Y - \mu)/\sigma$ denoted $N^*(t)$ is given by:

$$N^*(t) = \prod_{i=1}^{n} N^i\left(\frac{t}{\sigma}\right) \qquad \textbf{(EQ 11)}$$

*Stage 2:* We now consider the MGF $N^i(t/\sigma)$, which is given by $E(e^{Wt/\sigma})$. Expanding the exponential, we find that

$$N^i\left(\frac{t}{\sigma}\right) = E\left(e^{\frac{W^i t}{\sigma}}\right) = 1 + E(W^i)\frac{t}{\sigma} + \frac{E((W^i)^2)}{2!}\left(\frac{t}{\sigma}\right)^2 + \frac{E((W^i)^3)}{3!}\left(\frac{t}{\sigma}\right)^3 + \ldots \qquad \textbf{(EQ 12)}$$

Now, $E(W^i) = E(X^i\text{-}\mu^i) = E(X^i)\text{-}\mu^i = \mu^i\text{-}\mu^i = 0$, so we can ignore the second term in the expansion. Recall that $\sigma$ is the standard deviation of the sum of *n* random variables. When *n* is large, then $\sigma$ will also be large, which means that, to first order, we can ignore terms that have $\sigma^3$ and higher powers of $\sigma$ in the denominator in Equation 12. Therefore, for large *n*, we can write:

$$N^i\left(\frac{t}{\sigma}\right) \approx \left(1 + \frac{E((W^i)^2)}{2!}\left(\frac{t}{\sigma}\right)^2\right) = 1 + \frac{(\sigma^i)^2}{2}\left(\frac{t}{\sigma}\right)^2 \qquad \textbf{(EQ 13)}$$

where we have used the fact that $E((W^i)^2) = E((X^i\ \text{-}\mu_i)^2)$ = the variance of $X^i = (\sigma^i)^2$.

*Stage 3:* Returning to the expression in Equation 11, we find that

$$\log N^*(t) = \log\left(\prod_{i=1}^{n} N^i\left(\frac{t}{\sigma}\right)\right) = \sum_{i=1}^{n}\log\left(N^i\left(\frac{t}{\sigma}\right)\right) \approx \sum_{i=1}^{n}\log\left(1 + \frac{(\sigma^i)^2}{2}\left(\frac{t}{\sigma}\right)^2\right) \qquad \textbf{(EQ 14)}$$

It is easily shown by the Taylor series expansion that when *h* is small (so that $h^2$ and higher powers of *h* can be ignored) log(1+*h*) can be approximated by *h*. So, when *n* is large, and $\sigma$ is large, we can further approximate

$$\sum_{i=1}^{n}\log\left(1 + \frac{(\sigma^i)^2}{2}\left(\frac{t}{\sigma}\right)^2\right) \approx \sum_{i=1}^{n}\frac{(\sigma^i)^2}{2}\left(\frac{t}{\sigma}\right)^2 = \frac{1}{2}\left(\frac{t}{\sigma}\right)^2\sum_{i=1}^{n}(\sigma^i)^2 = \frac{1}{2}t^2 \qquad \textbf{(EQ 15)}$$

where, for the last simplification, we used Equation 10. Thus, log $N^*(t)$ is approximately 1/2 $t^2$, which means that

$$N^*(t) \approx e^{\frac{t^2}{2}} \qquad \textbf{(EQ 16)}$$

*Finale:* But this is just the MGF of a standard normal variable with zero mean and a variance of 1 (Equation 9). Therefore, $(Y - \mu)/\sigma$ is a standard normal variable, which means that $Y\sim N(\mu,\sigma^2)$. We have therefore shown that the sum of a large number of independent random variables is distributed as a normal variable whose mean is the sum of the individual means and whose variance is the sum of the individual variances (Equation 10), which, is, of course, the central limit theorem!

## 2.2 Sampling a population

The universe of individuals under study constitutes a *population* that can be characterized by its inherent *parameters* such as its range, minimum, maximum, mean, or variance. In many practical situations the population is infinite, so we have to estimate its parameters by studying a carefully chosen subset or *sample*. The parameters of a sample, such as its range, mean, and variance, are called its *statistics*. In standard notation, population parameters are denoted using the Greek alphabet and sample statistics are represented using the Roman alphabet. For example, the population mean and variance parameters are denoted $\mu$ and $\sigma^2$ respectively and the corresponding sample mean and variance statistics are denoted *m* (or $\bar{x}$) and $s^2$ respectively.

It is important to carefully identify the underlying population as the next example illustrates.

**Example 5: Choice of population**

Suppose that you capture a trace of all UDP packets sent on a link from your campus router to your university's Internet Service Provider from 6am to 9pm on Monday, November 17, 2008. What is the underlying population? There are many choices:

- The population of UDP packets sent from your campus router to your university's Internet Service provider from *12:00:01 am to 11:59:59pm* on November 17, 2008.
- The population of UDP packets sent from your campus router to your university's Internet Service provider from 12:00:01 am to 11:59:59pm *on Mondays*.
- The population of UDP packets sent from your campus router to your university's Internet Service provider from 12:00:01 am to 11:59:59pm *on days that are not holidays.*
- The population of UDP packets sent from your campus router to your university's Internet Service provider from 12:00:01 am to 11:59:59pm *on a typical day.*
- The population of UDP packets sent *from a typical university's campus router to a typical university's* Internet Service provider from 12:00:01 am to 11:59:59pm on a typical day.
- The population of UDP packets sent from a *typical access router to a typical ISP router* from 12:00:01 am to 11:59:59pm on a typical day.
- ...
- The population of all UDP packets sent on the Internet in 2008.
- The population of all UDP packets sent since 1969.

Each population in this list is a superset of the previous population. As you go down the list, therefore, conclusions drawn from your sample are more general. Simultaneously, the conclusions regarding the population that you draw from your sample are less valid. The difficulty is choosing an appropriate population in the spectrum between the most specific population (which is the sample itself) where your conclusions are certainly true and the most general population about which usually no valid conclusions can be drawn. Unfortunately, the only guide to making this judgement is experience and even experts may disagree with any decision you make.

[]

### 2.2.1 Types of sampling

As the previous example shows, first collecting a sample and then identifying the corresponding population puts the metaphorical cart in front of the horse. Instead, one should first identify a population to study and only then choose samples that are *representative* of that population. By representative, we mean a sample chosen such that every member of the population is equally likely to be a member of the sample. In contrast, if the sample is chosen so that some members of the population are more likely to be in the sample than others, then the sample is *biased* and the conclusions drawn from it may be inaccurate. Of course, representativeness is in the eye of the beholder. Nevertheless, explicitly stating the population and then the sampling technique will aid in identifying and removing hidden biases.

Here are some standard sampling techniques:

Random: In random or *proportional* sampling, an unbiased decision rule is used to select elements of the sample from the population. An example of such a rule 'Choose an element of the population with probability 0.05.' In running simulations, the choice of random seed values in random number generators perturbs simulation trajectories so that one can argue that the results of the simulation are randomly selected from the space of all possible simulation trajectories.

Stratified random: In this approach, the population is first categorized into groups of elements that are expected to differ in some significant way. Then, each group is randomly sampled to create a overall sample of the population. For example, one could first categorize packets on a link according to their transport protocol (TCP, UDP, or other), then sample each category separately in proportion to their ratio in the population. This way, if there are very few 'other' packets, the sample would still have enough elements to correctly estimate the population parameters for this packet type.

Systematic: This approach is similar to random sampling but sometimes simpler to carry out. We assume that the population can be enumerated in some random fashion (i.e., with no discernible pattern). Then, the systematic sampling rule is to select every *k*th element of this random enumeration. For instance, if we expected packet arrivals to a switch to be no particular order with respect to their destination port, then the destination port of every 100th arriving packet would constitute a systematic sample.

Cluster: Cluster sampling, like stratified sampling, is appropriate when the population naturally partitions itself into distinct groups. As with stratified sampling, the population is divided into groups and each group is separately sampled. Grouping may reflect geography or an element type. However, unlike stratified sampling, with cluster sampling the identity of the cluster is preserved, and statistics are computed individually for each cluster. In contrast to stratified sampling, where the grouping attempts to increase precision, with cluster sampling, the goal is to reduce the cost of creating the sample. Cluster sampling may be done hierarchically, with each level of the hierarchy or *stage* further refining the grouping.

Purposive: Here, the idea is to sample only elements that meet a specific definition of the population. For example, suppose we wanted to study all IP packets of length 40 bytes (corresponding to a zero data payload). Then, we could set up a packet filter that captured only these packets, constituting a purposive sample.

Convenience: A convenience sample involves studying the elements of the population that happen to be handy or conveniently available. For example, you may examine call traces from a cooperative cell phone operator to estimate mean call durations. Although it may not be possible to claim that call durations on that provider are representative of all cellular calls (because the duration is influenced by pricing policies of each operator), this may be all that is available and is certainly better than not having any data at all.

### 2.2.2 Scales

Gathering a sample requires measuring some physical quantity along a scale. Not all quantities correspond to values along a real line. In fact, we distinguish between four types of scales:

1. Nominal: A nominal scale corresponds to categories. Quantities arranged in a nominal scale cannot be mutually compared. For example, the transport-protocol type of a packet (i.e., UDP, TCP, other) constitutes a nominal scale.
2. Ordinal: An ordinal scale defines an ordering but distances along the ordinal scale are meaningless. A typical ordinal scale is the *Likert* scale, where 0 may correspond to ‘strongly disagree,’, 1 to ‘disagree,’, 2 to ‘neutral,’ 3 to ‘agree,’ and 4 to ‘strongly agree.’ A similar scale, with the scale ranging from ‘poor’ to ‘excellent’ is often used to compute the Mean Opinion Score (MOS) of a set of consumers of audio or video content to rank the quality of the content.
3. Interval: An interval scale defines an ordering where distances between indices are meaningful, but there is no absolute zero value. That is, the values are invariant to an affine scaling (multiplication by a constant followed by addition of another constant). A good example is vonNeumann-Morgenstern utilities.
4. Ratio: A ratio scale is an interval scale that also has a well-defined zero element, so that all indices are unique. Quantities such as packet length and inter-arrival time can be measured on ratio scales.

It is important to keep track of the type of scale corresponding to each measured quantity. A typical mistake is to assume that an ordinal scale can be treated as an interval or ratio scale.

### 2.2.3 Outliers

A common problem when collecting data is to find that some data elements are significantly out of line compared with the rest. These outliers can arise due to circumstances such as a failure in the measuring instrument or software test harness, overflowing counters, or abnormal circumstances.

Although ignoring outliers is common practice, there are two reasons for treating them with care. First, the presence of an outlier is often indicative of poor data collection practices. Often, examining the root cause of an outlier reveals problems with the entire measurement setup. Fixing these problems usually not only eliminates outliers but also results in the collection of statistically valid data. Second, outliers indicate the presence of unusual or unsuspected complexity in the operation of a system. Explaining outliers can reveal to deeper appreciation of the underlying system. Therefore, when collecting samples, it is imperative to pay special attention to outliers, making certain that these are truly statistical aberrations before dismissing them or removing them from the data set.

## *2.3 Describing a sample parsimoniously*

After gathering a sample, the next step is to describe it parsimoniously, that is, with a few well-chosen statistics. These statistics constitute a *model* of the sample, in that they represent it. Each data item in the sample can be viewed as arising partly from a model and partly from an error term, that is:

$$Data = Model + Error$$

A good model accounts for each element of the sample while minimizing the error. Naturally, the greater the number of parameters in a model, the better it fits the sample: the best model of a sample is the sample itself. However, a model with a hundred or a hundred million parameters provides no insight. Our goal is to describe as much of the data as possible with the least number of parameters. Here, we consider some well known descriptors of sample data.

### 2.3.1 Tables

The simplest technique to represent data is by tabulation. Let the *i*th sample value be denoted $x_i$ and let $n(x)$ denote the number of occurrences of the value $x$ in a sample. Then, a table is defined as the set of tuples $(x, n(x))$.

### 2.3.2 Bar graphs, Histograms, and Cumulative Histograms

Bar graphs and histograms graphically represent the number of occurrences of sample values (i.e., $n(x)$) as a function of $x$. When $x$ is measured on a nominal or ordinal scale, histograms and bar graphs both consist of a set of bars or rectangles of height proportional to $n(x)$ for each value of $x$. When $x$ is measured on an interval or a ratio scale, the scale is first divided into contiguous ranges called *bins*. Bins may differ in width, which is also called the *bin size*. If all bins are the same size then histograms and bar graphs are the same. However, if bin sizes differ, in a histogram the height of the bar is inversely proportional to the bin size but in a bar graph, the height is unchanged. In a bar graph only the height of the bar (rectangle) is significant, whereas for a histogram the area of the rectangle is significant. A histogram is, therefore, a quantized (or approximate) probability density function of the underlying population, becoming identical to it in the limit as the number of bins goes to infinity.

**Example 6: Bar graphs and histograms**

Consider the following set of observations in a sample:

| Data value | Frequency |
|---|---|
| 1 | 5 |
| 2 | 7 |
| 3 | 2 |
| 7 | 2 |
| 10 | 1 |

TABLE 1. **Sample data**

Note that the number of samples is quite sparse after the value 3. We have many choices of representation. For example, we can treat the data value as being measured on an ordinal scale and show the frequency of each data value found in the sample. This results in the bar graph/histogram show in Figure 1.

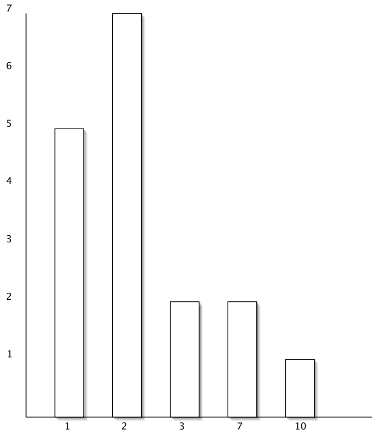


FIGURE 1. **Bar graph of the sample with data values on an ordinal scale**

We could also treat the data values as being on an interval scale, with bins [0.5,1.5], [1.5, 2.5], [2.5,4.5], [4.5, 10.5] where (a) the bin limits are chosen so that there can be no ambiguity as to which bin a value falls into and (b) the bins are not equally sized. This results in the bar graph and histogram shown in Figure 2.

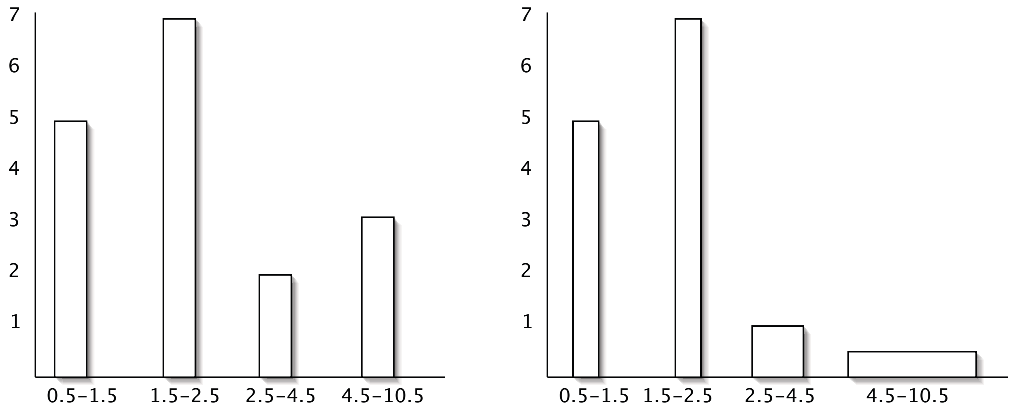


FIGURE 2. **Bar graph (left) and histogram (right) for the sample with data values on an interval scale**

[]

Choosing a bin size for variables measured on a interval or ratio scales requires choosing appropriate bin sizes. Unfortunately, this can usually only be accomplished by trial and error. If the bin sizes are too small, then many bins are likely to be empty and the histogram is visually too wide. On the other hand, if the bin sizes are too large, then all the data values may cluster into a handful of bins, hiding finer-scale structure. Several heuristics for bin sizing are known for the case of equally sized bins. For example, *Scott's choice* of bin width is $width = \frac{3.5s}{\sqrt[3]{n}}$ where $s$ is the standard deviation of the sample (i.e., the square root if its variance - see Section 2.3.5 on page 37).

The cumulative histogram of a sample is a histogram where the value represented by the *m*th bin is the count of sample data values up to an including those in the *m*th bin, i.e., $\sum_{i=1}^{m} n(x_i)$. This can be viewed as the quantized version (or approximation) of the cumulative density function of the underlying population.

### 2.3.3 The sample mean

The sample mean $\bar{x}$ of a sample with *n* elements is defined as

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i = \frac{1}{n}\sum_{i} x_i \qquad \text{(EQ 17)}$$

Alternatively, given a sample in tabular form, we can sum over the different possible values of *x*:

$$\bar{x} = \frac{1}{n}\sum_{x} n(x)x \qquad \text{(EQ 18)}$$

Adopting a frequentist approach to probability, where *P(x),* the probability of a value *x*, is defined as the limiting value of *n(x)/n*, we see that:

$$\lim_{n \to \infty} \bar{x} = \lim_{n \to \infty} \frac{1}{n}\sum_{x} n(x)x = \sum_{x} xP(x) = \mu \qquad \text{(EQ 19)}$$

This shows that as the sample size becomes very large, its mean is the expected value of a data item.

It turns out that the expected value of the sample mean *E($\bar{x}$)* is also μ (this is not the same as the limiting value of the mean - why?) To see this, we start with the definition of the sample mean: $\bar{x} = \frac{1}{n}(x_1 + x_2 + \ldots + x_n)$, so that $n\bar{x} = (x_1 + x_2 + \ldots + x_n)$. Taking expectations of both sides,

$$E(n\bar{x}) = E(x_1 + x_2 + \ldots + x_n) \qquad \text{(EQ 20)}$$

From the sum rule of expectations, we can rewrite this as:

$$E(n\bar{x}) = E(x_1) + E(x_2) + \ldots + E(x_n) = n\mu \qquad \text{(EQ 21)}$$

Therefore,

$$E(\bar{x}) = E\left(\frac{n\bar{x}}{n}\right) = \frac{E(n\bar{x})}{n} = \mu \qquad \text{(EQ 22)}$$

The mean of a sample is a good representative of the sample for two reasons. First, for a finite sample size, the mean is an *estimator* of the population mean. An estimator is *unbiased* if its expected value is the corresponding population parameter. From Equation 19, the mean is an unbiased estimator of the population mean. It turns out that if a population is normally distributed, then the mean is also the most *efficient* unbiased estimator of the population mean, in that it has the least variance of all unbiased estimators of the population mean. Second, it can be easily shown that the mean value of a sample is the value of $x^*$ that minimizes $\sum_{i=1}^{n}(x_i - x^*)^2$ (i.e., the sum of squared deviations from $x^*$, which can be interpreted as errors in choosing $x^*$ as a representative of the sample). In this sense, the mean is the 'central' value of a sample.

The mean is therefore the most widely used first-order descriptor of a population. Nevertheless, when using the mean of a sample as its representative, the following issues must be kept in mind:

- The mean of a sample may significantly differ from the true population mean. Consider the means of *m* samples, each with $n_m$ data items (Figure 3). These means are likely to differ in value and can be thought of as themselves being data items in a sample with *m* elements, drawn from a population of sample means. The distribution of this population of sample means is called the *sampling distribution of the mean*. If this distribution has a large variance, then the mean of any particular sample may be far from representative of the population mean. Therefore, the mean of a sample, especially when the sample size is small, should be treated only as a rough guide to the truth. We will examine the topic of statistical significance of the mean of a sample in a later section.

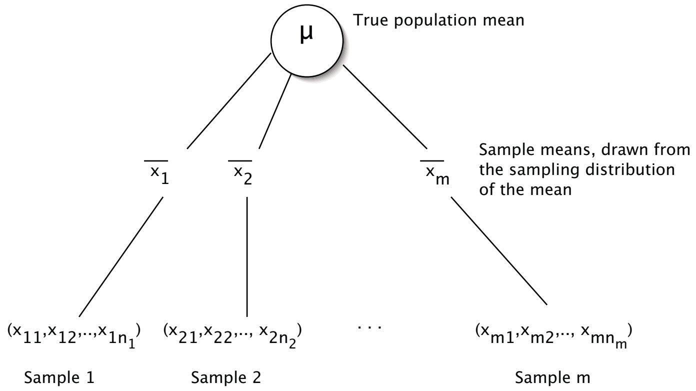

μ
True population mean
$\overline{x_1}$
$\overline{x_2}$
$\overline{x_m}$
Sample means, drawn from the sampling distribution of the mean
$(x_{11}, x_{12}, ..., x_{1n_1})$
$(x_{21}, x_{22}, ..., x_{2n_2})$
. . .
$(x_{m1}, x_{m2}, ..., x_{mn_m})$
Sample 1
Sample 2
Sample m

FIGURE 3. **Sampling distribution of the mean**

- The mean of a sample can be greatly influenced by outliers. Imagine a system where most packet interarrival times are small but there is one very large gap between packet bursts, corresponding to a very large interarrival time. Unless this outlier is excluded, this single value can bias the mean interarrival time.

**Example 7: Outliers**

Consider the following sample:

TABLE 2. **A sample with an outlier**

| Data value | Frequency |
|---|---|
| 1 | 5 |
| 2 | 7 |
| 3 | 2 |
| 7 | 2 |
| 1000 | 1 |

The sample mean excluding the last value is (1*5 + 2*7 + 3*2 + 7*2)/(5+7+2+2) = 2.43. The sample mean including the last value is 61.1. It is hard to argue that the mean is representative of this sample, due to the influence of the outlier.

[]

- The mean of a multi-modal distribution is not particularly representative of the data. Consider a link where interarrival times are either small (< 1ms) or large (> 10s). This will result in a bi-modal distribution of sample values. The mean of any sample is therefore not a good representative of a given sample. In such cases, it is best to cluster the sample and compute the means for each cluster separately.

The variance of the sample mean (that is, the variance of the sampling distribution of the mean) can be computed as follows. Recall that $n\bar{x} = (x_1 + x_2 + \ldots + x_n)$. Taking the variance of both sides,

$$V(n\bar{x}) = V(x_1 + x_2 + \ldots + x_n) \quad \textbf{(EQ 23)}$$

Now, the variance of a sum of a set of independent random variables is the sum of their variances. If we assume that each data value in a sample is independent (an assumption that may not always hold true), then

$$V(n\bar{x}) = V(x_1) + V(x_2) + \ldots + V(x_n) = n\sigma^2 \quad \textbf{(EQ 24)}$$

Therefore,

$$V(\bar{x}) = V\left(\frac{n\bar{x}}{n}\right) = \frac{V(n\bar{x})}{n^2} = \frac{\sigma^2}{n} \quad \textbf{(EQ 25)}$$

Therefore, the variance of the sample mean is 1/*n* of the variance of the population variance. In other words, as the size of a sample increases, the sample mean (whose expected value is μ) has a smaller and smaller variance, and the mean of each sample will be tightly clustered around μ.

### 2.3.4 The sample median

The median value of a sample is the value such that 50% of the samples are larger than this value. For a sample with an odd number of elements, it is the middle element after sorting. For a sample with an even number of elements, it is the mean of the two middle elements after sorting.

The median is a better representative of the mean for samples that contain outliers, in that it is relatively insensitive to outliers. It is also an unbiased estimator of the population mean. However, it can be shown that if the underlying distribution is normal, then the asymptotic variance of the median of a sample is 1.57 times larger than the asymptotic variance of the sample mean. Hence, if the underlying distribution is normal, the same accuracy in estimating the population mean can be obtained by collecting 100 observations and computing their mean or by collecting 157 samples and computing their median. If the underlying distribution is unimodal and sharper-peaked than normal (also called *leptokurtic*), then the median is a more efficient estimator than the mean, because, in such situations, the variance of the mean is higher than the variance of the median due to the presence of outliers.

### 2.3.5 Measures of variability

Unlike the mean or the median, which seek to represent the ‘central tendency’ of a sample, we now consider ways of representing the degree to which the data values in a sample differ from each other. These are also called ‘measures of variability.’

The simplest measure of variability is the *range*, which is the difference between the largest and smallest value. The range is susceptible to outliers and therefore not reliable. A better measure is to sort the data values and then determine the data values that lie at *q*% and 1-*q*%. The difference between the two values is the range of values in which the central 1-2*q*% of the sample lies. This conveys nearly the same information as the range but with less sensitivity to outliers. A typical value of *q* is 25, in which case this measure is also called the *inter-quartile range*. In the context of delay bounds and service-level agreements, a typical value of *q* is 5 (so that the span is 5%-95%).

**Example 8: Inter-quartile range**

Consider the sample in Table 2. There are 17 data values, so the 25th percentile index is the fourth one, and the 75th percentile index is the 13th one. The fourth value in sorted order is 1 and the 13th value is 3. Hence, the inter-quartile range is 2.

[]

Although ranges convey some information, they do not tell us what fraction of the data values are clustered around the sample mean. This information can be represented by the sample variance $m_2$ which is defined as:

$$m_2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2 = \frac{1}{n}\sum_{x}^{n}(x - \bar{x})^2 n(x) \qquad \textbf{(EQ 26)}$$

Clearly, the variance increases if the sample values are distant from the mean (so that the mean is a poor representative of the sample) and is zero if all the data values are exactly clustered at the mean (in which case the mean perfectly represents the sample). The positive square root of the variance is called the *standard deviation.* Unlike the variance, the standard deviation has the same units as the data values in the sample.

A simple technique to compute the variance of a sample is to maintain a running total of three quantities *n*, $\sum_i x_i$, and $\sum_i x_i^2$.

Then, the variance can be computed as:

$$m_2 = \frac{1}{n}\left(\sum_i x_i^2 - \frac{\left(\sum_i x_i\right)^2}{n}\right) \qquad \textbf{(EQ 27)}$$

In the same way that the sample mean estimates the population mean, the sample variance is an estimator for the population variance, i.e. $E(X-\mu)^2$. However, the sample variance is *not* an unbiased estimator of the population variance--it is slightly smaller--with $E(m_2) = (n-1)\sigma^2/n$.

To prove this, recall that each element $x_i$ in a sample can be thought of as being a random variable whose distribution is identical to that of the population, with an expected value of μ and a variance of $\sigma^2$. That is, $E(x_i) = \mu$ and $V(x_i) = E((x_i\text{-}\mu)^2) = \sigma^2$. Now, by definition of $m_2$:

$$m_2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2 = \frac{1}{n}\left(\sum_{i=1}^{n}(x_i - \mu)^2 - n(\bar{x} - \mu)^2\right) \qquad \textbf{(EQ 28)}$$

where the second step can be verified by expansion. Taking expectations on both sides, we find that:

$$E(m_2) = \frac{1}{n}E\left(\sum_{i=1}^{n}(x_i - \mu)^2 - n(\bar{x} - \mu)^2\right) = \frac{1}{n}\left(E\left(\sum_{i=1}^{n}(x_i - \mu)^2\right) - nE((\bar{x} - \mu)^2)\right) \qquad \textbf{(EQ 29)}$$

Now, $E\left(\sum_{i=1}^{n}(x_i - \mu)^2\right) = \sum_{i=1}^{n}E(x_i - \mu)^2 = \sum_{i=1}^{n}\sigma^2 = n\sigma^2$. Also, because $E(\bar{x}) = \mu$, $E((\bar{x} - \mu)^2) = V(\bar{x}) = \frac{\sigma^2}{n}$. Substituting these into Equation , we find that

$$E(m_2) = \frac{1}{n}(n\sigma^2 - \sigma^2) = \frac{(n-1)}{n}\sigma^2 \qquad \textbf{(EQ 30)}$$

as stated. Therefore, to obtain an unbiased estimate of $\sigma^2$, we should multiply $m_2$ by $n/(n-1)$, which amounts to computing $\frac{1}{n-1}\left(\sum_{i=1}^{n}(x_i - \bar{x})^2\right)$.

## 2.4 Inferring population parameters from sample parameters

Thus far we have focused on statistics that describe a sample in various ways. A sample, however, is usually only a subset of the population. Given the statistics of a sample, what can we infer about the corresponding population parameters? If the sample is small or if the population is intrinsically highly variable, then there is not much we can say about the population. However, if the sample is large, there is reason to hope that the sample statistics are a good approximation to the population parameters. We now quantify this intuition.

Our point of departure is the central limit theorem, which states that the sum of the *n* independent random variables, for large *n*, is approximately normally distributed. Suppose we collect a series of *m* samples, each with *n* elements, from some population. (In the rest of the discussion we will assume that *n* is large enough that the central limit theorem applies.) If the elements of each sample are independently and randomly selected from the population, we can treat the sum of the elements of each sample as the sum of *n* independent and identically distributed random variables $X_1, X_{2,...}, X_n$. That is, the first element of the sample is the value assumed by the random variable $X_1$, the second element is the value assumed by the random variable $X_2$, and so on. From the central limit theorem, the sum of these random variables is normally distributed. The mean of each sample is the sum divided by a constant, so the mean of each sample is also normally distributed. This fact allows us to determine a range of values where, with high confidence, the population mean can be expected to lie.

To make this more concrete, refer to Figure 3 and consider sample 1. The mean of this sample is $\overline{x_1} = \frac{1}{n}\sum_i x_{1i}$. Similarly $\overline{x_2} = \frac{1}{n}\sum_i x_{2i}$, and, in general, $\overline{x_k} = \frac{1}{n}\sum_i x_{ki}$. Define the random variable $\overline{X}$ as taking on the values $\overline{x_1}, \overline{x_2}, ..., \overline{x_n}$. The distribution of $\overline{X}$ is called the *sampling distribution of the mean*. From the central limit theorem, $\overline{X}$ is approximately normally distributed. Moreover, if the elements are drawn from a population with mean μ and variance $\sigma^2$, we have already seen that $E(\overline{X}) = \mu$ (Equation 22) and $V(\overline{X}) = \sigma^2/n$ (Equation 25). These are, therefore, the parameters of the corresponding normal distribution, i.e., $\overline{X} \sim N(\mu, \sigma^2/n)$. Of course, we do not know the true values of μ and $\sigma^2$.

If we knew $\sigma^2$, we can estimate a range of values in which μ will lie, with high probability, as follows. For any normally distributed random variable $Y \sim (\mu_Y, \sigma_Y^2)$, we know that 95% of the probability mass lies within 1.96 times the standard deviations of its mean, and 99% of the probability mass lies within 2.576 standard deviations of its mean. So, for any value *y*:

$$P(\mu_Y - 1.96\ \sigma_Y < y < \mu_Y + 1.96\ \sigma_Y) = 0.95 \quad \textbf{(EQ 31)}$$

The left and right endpoints of this range are called the *critical values* at the 95% confidence level: an observation will lie beyond the critical value, assuming that the true mean is $\mu_Y$, in less than 5% (or 1%) of observed samples. This can be rewritten as:

$$P(|\mu_Y - y| < 1.96\ \sigma_Y) = 0.95 \quad \textbf{(EQ 32)}$$

Therefore, from symmetry of the absolute value:

$$P(y - 1.96\ \sigma_Y < \mu_Y < y + 1.96\ \sigma_Y) = 0.95 \quad \textbf{(EQ 33)}$$

In other words, given any value *y* drawn from a normal distribution whose mean is $\mu_Y$, we can estimate a range of values where $\mu_Y$ must lie with high probability (i.e. 95% or 99%). This is called the *confidence interval* for $\mu_Y$.

We just saw that $\overline{X} \sim N(\mu, \sigma^2/n)$. Therefore, given the sample mean $\overline{x}$:

$$P\left(\overline{x} - 1.96\ \frac{\sigma}{\sqrt{n}} < \mu < \overline{x} + 1.96\ \frac{\sigma}{\sqrt{n}}\right) = 0.95 \quad \textbf{(EQ 34)}$$

and

$$P(\bar{x} - 2.576 \frac{\sigma}{\sqrt{n}} < \mu < \bar{x} + 2.576 \frac{\sigma}{\sqrt{n}}) = 0.99 \qquad \text{(EQ 35)}$$

This allows us, assuming we knew $\sigma^2$, to compute the range of values where the population mean will lie, which 95% or 99% confidence, given the sample mean.

Note that a confidence interval is constructed from the observations in such a way that there is a known probability, such as 95% or 99%, of it containing the population parameter of interest. It is not the population parameter that is the random variable - the interval itself is the random variable.

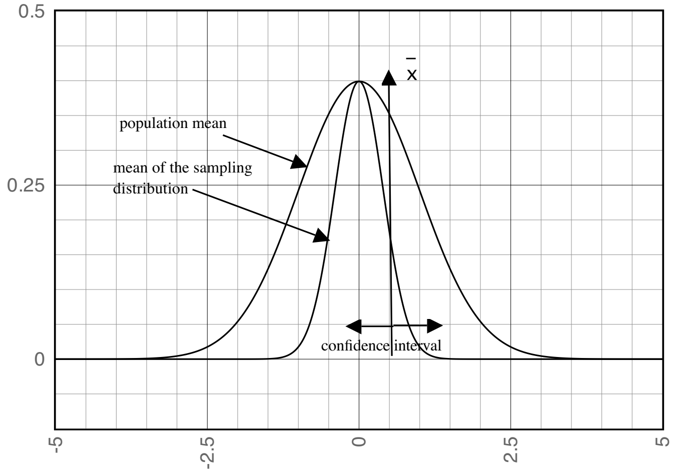


**FIGURE 4.** **Population and sample mean distributions**

The situation is graphically illustrated in Figure 4. Here, we assume that the population is normally distributed with mean 0. The variance of the sampling distribution (i.e. of $\bar{X}$) is $\sigma^2/n$, so it has a narrower spread than the population (with the spread decreasing as we increase the number of elements in the sample). A randomly chosen sample happens to have a mean of 0.5. This mean is the value assumed by a random variable $\bar{X}$ whose distribution is the sampling distribution of the mean. The double headed arrow around $\bar{x}$ indicates a confidence interval around it, in which, with high probability, μ must lie.

In almost all practical situations, we do not know $\sigma^2$. But all is not lost! Recall that an unbiased estimator for $\sigma^2$ is $m_2(n/(n-1)) = \frac{1}{n-1}\left(\sum_{i=1}^{n}(x_i - \bar{x})^2\right)$ (Equation 30). Therefore, assuming that this estimator is of good quality (in practice, this is true when $n >$ ~20), $\bar{X} \sim N(\mu, \frac{1}{n(n-1)}\left(\sum_{i=1}^{n}(x_i - \bar{x})^2\right))$. Therefore, when $n$ is sufficiently large, we can still compute the confidence interval in which the population mean lies with high probability.

**Example 9: (Confidence intervals)**

Consider the data values in Table 1 on page 34. What is the confidence interval in which the population mean lies?

We will temporarily ignore the fact that $n = 17 < 20$, so the central limit theorem is not likely to apply. The sample mean is 2.88. We compute $\sum_{i=1}^{n}(x_i - \bar{x})^2$ as 107.76. Therefore, the variance of the sampling distribution of the mean is estimated as 107.76/(17*16) = 0.396 and the standard deviation of this distribution is estimated as its square root, i.e., 0.63. Using the value of +/- 1.96σ for the 95% confidence interval and +/- 2.576σ for the 99% confidence interval, the 95% confidence interval is [2.88-1.96*0.63, 2.88+1.96*0.63] = [1.65, 4.11] and the 99% confidence interval is [1.26, 4.5].

[].

Because $\bar{x}$ is normally distributed with mean μ and variance $\sigma^2/n$, $\frac{(\bar{x}-\mu)}{\left(\frac{\sigma}{\sqrt{n}}\right)}$ is a *N(0,1)* variable, also called the *standard Z variable*. In practice, when $n > 20$, we can substitute $m_2(n/(n-1))$ as an estimate for $\sigma^2$ when computing the standard *Z* variable.

So far, we have assumed that *n* is large, so that the central limit theorem applies. In particular, we have made the simplifying assumption that the estimated variance of the sampling distribution of the mean is identical to the actual variance of the sampling distribution. When *n* is small, this can lead to underestimating the variance of this distribution. To correct for this, we have to re-consider the random variable $\frac{(\bar{x}-\mu)}{\left(\frac{\sigma}{\sqrt{n}}\right)}$ , which we estimate as the random variable $\frac{(\bar{x}-\mu)}{\sqrt{\frac{\sum_i (x_i-\bar{x})^2}{n(n-1)}}}$. The latter variable is called the *standard t variable*. Such variables obey the *t* distribution with *n*-1 *degrees of freedom* (a parameter of the distribution). The salient feature of the t distribution is that, unlike the normal distribution, its shape varies with the degrees of freedom, with its shape for $n > 20$ becoming nearly identical to the normal distribution.

How does this affect the computation of confidence intervals? Given a sample, we proceed to compute the estimate of the mean as $\bar{x}$ as before. However, to compute the, say, 95% confidence interval, we need to change our procedure slightly. We have to find the range of values such that the probability mass under the *t* distribution (not the normal distribution) centered at that mean and with variance $\frac{\sum_i (x_i-\bar{x})^2}{n(n-1)}$ is 0.95. Given the degrees of freedom (which is simply *n*-1), we can look this up in a standard *t* table. Then, we can state with 95% confidence that the population mean lies in this range.

**Example 10: (Confidence intervals for small samples)**

Continuing with the sample in Table 1 on page 34, we will now use the *t* distribution to compute confidence intervals. The unbiased estimate of the population standard deviation is 0.63. *n*=17, so this corresponds to a *t* distribution with 16 degrees of freedom. We find from the standard *t* table that a (0,1) *t* variable reaches the 0.025 probability level at 2.12, so that there is 0.05 probability mass beyond 2.12 times the standard deviation on both sides of the mean. Therefore, the 95% confidence interval is [2.88 -2.12*0.63, 2.88+2.12*0.63] = [1.54,4.22]. Compare this to the 95% interval obtained using the normal distribution. Similarly, the *t* distribution reaches the 0.005 probability level at 2.921, leading to the 99% confidence interval of [2.88 - 2.921 *0.63, 2.88+2.921 *0.63] = [1.03, 4.72].

[]

So far, we have focussed on estimating the population mean and variance and have computed the range of values in which the mean is expected to lie with high probability. These are obtained by studying the sampling distribution of the mean. We can obtain corresponding confidence intervals for the population variance by studying the *sampling distribution of the variance*. It can be shown that if the population is normally distributed, this sampling distribution is the $\chi^2$ distribution (discussed in more detail below). However, this confidence interval is rarely derived in practice, and so we will omit the details of this result.

## *2.5 Testing hypotheses about outcomes of experiments*

We often need to study the outcomes of experiments conducted to measure the performance of computer systems. We typically would either like to assert that a metric associated with the system has a certain value (such as having a mean value of

1.5 units), or that a new heuristic or algorithm improves the performance of the system. Here, we study techniques for making statistically valid assertions about the outcomes of experiments where we compare at most two values: we will study outcomes involving more than two experiments in 2.7 on page 58.

### 2.5.1 Hypothesis testing

Assertions about outcomes of an experiment can usually be re-formulated in terms of testing a *hypothesis*: a speculative claim about the outcome of an experiment. The goal of an experiment is to either show that the hypothesis is unlikely to be true (i.e., we can reject the hypothesis), or to show that the experiment is consistent with the hypothesis (i.e., the hypothesis need not be rejected).

This last statement bears some analysis. Suppose we are asked to check whether a coin is biased. We will start with the tentative hypothesis that the coin is unbiased, that is, P(heads) = P(tails) = 0.5. Then we toss the coin three times. Suppose we get three heads in a row. What does this say about our hypothesis? Conditional on the hypothesis being true, we have a probability of 0.5*0.5*0.5 = 12.5% that we obtain the observed outcome. This is not too unlikely, so perhaps the three heads in a row were simply due to chance. At this point, all we can state is that the experimental outcome is consistent with the hypothesis.

Now, suppose we flip the coin 10 times and see that it comes up heads nine times. If our hypothesis were true, then the probability of getting nine heads in 10 coin flips is given by the binomial distribution as $\binom{10}{1}0.5^9 0.5^1 = 10*0.5^{10}=10/1024 < 1\%$. Thus, if the hypothesis were true, this outcome if fairly unlikely (setting the bar for ‘unlikeliness’ at 1%). This is typically stated as: “we reject the hypothesis at the 1% confidence level.”

The probability of an outcome conditional on a hypothesis being true is called its *p-value.* If the outcome of an experiment has a *p*-value less than 1% (or 5%), then we would interpret the experiment as grounds for rejecting a hypothesis at the 1% (or 5%) level.

It is important to realize that the non-rejection of a hypothesis does not mean that the hypothesis is valid. For example, we could have made the hypothesis that the coin was biased, with P(heads) = 0.9. If we toss the coin three times and get three heads, the probability of that event, assuming the hypothesis were true, would be 0.9*0.9*0.9 = 0.73. So we cannot reject the hypothesis that the coin is biased. Indeed, with such a small number of experiments, we cannot invalidate an infinite number of mutually incompatible hypotheses!

We are therefore led to two inescapable conclusions. First, even the most careful experiment may lead to an incorrect conclusion due to random errors. Such errors may result in rejection of a hypothesis, event though it ought not be rejected, or in non-rejection, when it ought to. Second, the outcome of an experiment cannot result in the acceptance of a hypothesis, but can only reject or not reject it. We deal with each conclusion in turn.

### 2.5.2 Errors in hypothesis testing

Testing a hypothesis can never be entirely accurate. Random fluctuations in the outcomes of experiments may lead to non-rejection of a hypothesis when it should be rejected and rejecting it when it should not. We now discuss these errors in hypothesis testing.

Consider two universes in each of which a particular hypothesis is either valid or invalid. In each universe, we can expect one of two results from hypothesis testing: “The hypothesis is not rejected,” or “The hypothesis is rejected.” The four possible outcomes of testing are represented in the following table:

| | **Outcome of the experiment** | |
|---|---|---|
| **State of the universe** | *Reject hypothesis* | *Do not reject hypothesis* |
| *Hypothesis is invalid* | Good outcome $C_{00}$ | Bad outcome $C_{01}$<br>False negative or Type II error |
| *Hypothesis is valid* | Bad outcome $C_{10}$<br>False positive or Type I error | Good outcome $C_{11}$ |

- If the hypothesis is invalid and is rejected, then we have a good outcome. The probability of this event is denoted $C_{00}$.
- If the hypothesis is invalid but is not rejected, then we have a bad outcome. The probability of this event is denoted $C_{01}$.
- If the hypothesis is valid and is not rejected, then we have a good outcome. The probability of this event is denoted $C_{11}$.
- If the hypothesis is valid but is rejected, then we have a bad outcome. The probability of this event is denoted $C_{10}$.

We can use the $C_{ij}$s to define the following quantities:

| Term | Definition | Meaning |
|---|---|---|
| Concordance | $C_{11}+C_{00}$ | The probability of an accurate prediction |
| Error rate | $C_{10}+C_{01}$ | The probability of an inaccurate prediction |
| Sensitivity | $C_{11}/(C_{11}+C_{01})$ | Ability to predict correctly conditional on the hypothesis actually being valid |
| Specificity | $C_{00}/(C_{10}+C_{00})$ | Ability to eliminate a false hypothesis conditional on the hypothesis actually being invalid. |

These apply to all types of hypothesis testing. Our goal is to design experiments that maximize good outcomes while minimizing bad outcomes. In certain cases, we may trade off a higher sensitivity for a higher error rate, or a higher specificity for a lower concordance. A common rule is to limit Type I errors to 5%. That is, if the hypothesis is valid, we should not mistakenly reject it more than 5% of the time.

### 2.5.3 Formulating a hypothesis

We now return to the problem that an experiment can only result in rejection or non-rejection of a hypothesis. Therefore, we may end up not rejecting an invalid hypothesis. What guidelines should we use in choosing a hypothesis?

The standard technique is to formulate a *null hypothesis* that we believe is sufficient to explain the data unless statistical evidence strongly indicates otherwise. The null hypothesis should be formulated conservatively, that is, preserving the *status quo*, where this is applicable. A good way to think about this is in terms of a criminal trial. The judicial system starts with the presumption of innocence. It is up to the prosecution to prove that the defendant is guilty. If the prosecution cannot prove beyond reasonable doubt that the defendant is guilty, then the defendant is released. No doubt, this will let some guilty parties go unpunished. But it is preferable to the alternative, where the defendant is assumed guilty and must prove innocence.

In formulating a null hypothesis, it is necessary to be precise. In the words of Sir R.A. Fisher, the inventor of this approach, the null hypothesis should be “free from vagueness and ambiguity.” Otherwise, it is may be impossible to reject it, making our effort fruitless. Moreover, a hypothesis should be about a population parameter, not a sample (unless the sample includes the entire population).

**Example 10: (Formulating a null hypothesis)**

Consider a router on which we can execute either scheduling algorithm A or scheduling algorithm B. Suppose our goal is to show that scheduling algorithm A is superior to scheduling algorithm B for some metric. An acceptable conservative null hypothesis would be “Scheduling algorithm A and scheduling algorithm B have identical performance.” Given this assumption, we would expect that the performance metrics for both scheduling algorithms to be roughly the same (i.e., this is our expectation on the states of the world). If our experiments show this to be the case, for example, if the sample means of the performance metrics for both scheduling algorithms were nearly identical, then we would conclude that we do not have sufficient evidence to prove that scheduling algorithm B improved the system, a conservative and scientifically valid decision. In contrast, if the sample mean for algorithm A were much higher than the sample mean for algorithm B (we will quantify this shortly), then the experiment would be inconsistent with our null hypothesis, and we would reject it, giving credence to the belief that scheduling algorithm A was indeed better[1].

[1] In the Bayesian formulation of hypothesis testing, we view the experiment as updating a prior expectation on the state of the world, thus refining our model for the state of the world. Here, we are presenting the classical statistical view on hypothesis testing.

If we were to invert the null hypothesis, for example stating it as “Scheduling algorithm A is better than scheduling algorithm B.” then by being unable to reject the null hypothesis, we may come to an unwarranted conclusion. In any case, this hypothesis is imprecise, in that we did not quantify *how* much better algorithm A is supposed to be better than scheduling algorithm B, so ought to be deprecated on those grounds alone.

[]

We represent the null hypothesis using the symbol $H_0$. Alternatives to the null hypothesis are usually labelled $H_1$, $H_2$, etc. The steps in hypothesis testing depend on whether the outcome of an experiment is being compared with a fixed quantity, or with the outcome of another quantity. In the next sub-section, we will consider outcomes that are compared with a fixed quantity, deferring comparison of outcomes of two experiments to 2.5.5 on page 45.

Hypotheses can be ‘two-tailed’ or ‘one-tailed.’ We reject a two-tailed hypothesis if the sample statistic significantly differs from the conjectured corresponding population parameter in absolute value. For example, suppose we hypothesize that the population mean $\mu$ is 0. If observations indicate that $|\bar{x}| > a$, where $a$ is the *critical value,* then we reject the hypothesis. In contrast, we reject a one-tailed hypothesis if the sample statistic significantly differs from the corresponding population parameter in a pre-specified direction (i.e., is smaller than or larger then the conjectured population parameter), where experimental conditions allow us to rule out deviations in the other direction. An example of a one-tailed hypothesis is $\mu < 0$. If $\bar{x} > a$, where $a$, again, is the critical value, we can reject this hypothesis. Note that we do not consider the ‘other tail,’ that is, the possibility that $\bar{x} < a$ (why?).

### 2.5.4 Comparing an outcome with a fixed quantity

To fix ideas, suppose, based on some physical considerations, we expect the mean of a population to be some value, say 0. Assume that there is no particular *status quo* that we are trying to maintain. Therefore, a reasonable null hypothesis is:

$$H_0\text{: the population mean is } 0$$

To test this hypothesis, we run a series of experiments to collect observations and compute the sample mean. Assuming for now that the number of samples is large, the mean will be drawn from a normal distribution and we can use this fact to compute its confidence interval (say at the 99% level) using the techniques in Section 2.4 on page 39. We then check if 0 lies within this interval. One of two cases arise:

- 0 lies in the 95% (99%) confidence interval of the sample mean. In this case, we cannot reject the null hypothesis. This is usually interpreted to mean that with 95% (99%) confidence the population mean is indeed 0. Of course, all have have shown is that the outcome of this experiment has a likelihood greater than 95% (99%) conditional on the mean being 0 (i.e., it is consistent with the null hypothesis).
- 0 does not lie in the 95% (99%) confidence interval of the sample mean. In this case, we reject the null hypothesis. This is usually interpreted to mean that, with high confidence, the population mean is not 0. Again, all we have shown is that, conditional on the mean being 0, the outcome we saw was rather unlikely, so we have good reason to be suspicious of the null hypothesis.

This example is easily generalized. Suppose we want to establish that the population mean is $\mu_0$. We compute the sample mean $\bar{x}$ as before. Then, we test the hypothesis:

$$H_0\text{: } (\bar{x} - \mu_0) = 0$$

which can be tested as described above, with identical conclusions being drawn about the results[2].

**Example 11: (Testing for a zero mean)**

[2] Advanced readers will note that in this section, we have switched from the Fisher to the Neyman-Pearson approach to hypothesis testing. This hybrid approach is widely used in modern scientific practice.

Returning to Example 10, note that the 99% confidence interval for the mean of the sample data, using the *t* test, was [1.03, 4.72]. Therefore, we can state with 99% confidence (with the caveats stated above) that the mean of the underlying population is not 0.

[]

### 2.5.5 Comparing outcomes from two experiments

Suppose we want to test the hypothesis that two samples are drawn from different populations. To fix ideas, consider the situation where we are comparing two systems—a system currently in use and a system that incorporates a new algorithm—on the basis of a particular performance metric. We assume that we can collect performance metrics from each system multiple times to obtain two samples. If the systems do not differ to a statistically significant degree, then both samples would be drawn from the same underlying population, with, for example, the same population mean, and therefore would have similar statistics, such as the sample mean. However, if the statistics are significantly different, then we infer that the two samples are likely to be drawn from different populations and the new algorithm does indeed affect the performance of the system.

The null hypothesis is the statement:

$H_0$*: the two systems are identical*

We reject $H_0$ if it is sufficiently unlikely that the two samples are drawn from the same population because this will result in the conservative position that the new algorithm does not improve the performance of the system.

Suppose we collect *n* results from the first system, labelled A, to get sample values $a_1, a_2,...,a_n$ and collect *m* results from the second system, labelled B, to get measurements $b_1, b_2,...,b_m$. Let the means of these results be $\overline{a}$ and $\overline{b}$ with sample variances $m_2(a)$ and $m_2(b)$.

If $n=m$, then we define an auxiliary random variable C=A-B, which takes values $c_i = a_i - b_i$. Then, we redefine the hypothesis as:

$H_0$*: the population mean of C is zero*

This can be easily tested using the approach described in 2.5.4 on page 44.

**Example 12: Comparing two samples**

Suppose that you are using simulations to study the effect of buffer size at some network queue on packet loss rate. You would like to see if increasing the buffer size from 5 packets to 100 packets has a significant effect on loss rate. To do so, suppose that you run 10 simulations for each buffer size, resulting in loss rates shown below:

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Loss rate with 5 buffers** | 1.20% | 2.30% | 1.90% | 2.40% | 3.00% | 1.80% | 2.10% | 3.20% | 4.50% | 2.20% |
| **Loss rate with 100 buffers** | 0.10% | 0.60% | 1.10% | 0.80% | 1.20% | 0.30% | 0.70% | 1.90% | 0.20% | 1.20% |

Does the buffer size have a significant effect on loss rate?

Denoting by $a_i$ the loss rate with 5 buffers and by $b_i$ the loss rate with 100 buffer, we define the auxiliary variable $c = a-b$ that takes values (1.2-0.1), (2.3-0.6),..., (2.2-1.2), so that *c* is given by:

$c$ = {1.1, 1.7, 0.8, 1.6, 1.8, 1.5, 1.4, 1.3, 4.3, 1.0}

We compute the mean as 1.65 and sample variance $m_2$ as 0.87, so that the unbiased estimator for the population variance is given by $(n/n-1)m_{2,}$ = 0.97. The variance of the sample mean is given by $m_2/n$ = 0.097, corresponding to a standard deviation of 0.31. Because the number of values is smaller than 20, we use the *t* distribution with 9 degrees of freedom to compute the confidence interval at the 95% level as 1.65±0.70. This interval does not include 0. Thus we conclude that the change in the buffer size does significantly affect the loss rate.

[]

If $n \neq m$ , the situation is somewhat more complex. We first use $m_2(a)$ to compute the confidence interval for A's performance metric around its sample mean $\bar{a}$ and similarly use $m_2(b)$ to compute the confidence interval for B's performance metric around its sample mean $\bar{b}$ (using the normal or t distribution, as appropriate). Now, one of the following two cases hold:

- The confidence intervals do not overlap. Recall that with 95% (or 99%) confidence, A's and B's population means lie within the computed confidence intervals. If the null hypothesis were true and the population means coincided, then it must be the case that either A's population mean or B's population mean lies outside of its computed confidence interval. However, this has a probability lower than 5% (1%). Therefore, we reject the hypothesis.
- The confidence intervals overlap. In this case, there is some chance that the samples are drawn from the same population. The next steps depend on whether we can make one of two assumptions: (1) the sample variances are the same or (2) *n* and *m* are both large.

(1) *Sample variances are the same*. In this case, we define the auxiliary variable *s* by:

$$s^2 = \frac{\sum_{i=1}^{n}(a_i - \bar{a})^2 + \sum_{i=1}^{m}(b_i - \bar{b})^2}{m + n - 2} \qquad \textbf{(EQ 36)}$$

Then, it can be shown that if the two samples are drawn from the same population the variable *c* defined by:

$$c = \frac{\bar{a} - \bar{b}}{s\sqrt{\frac{1}{m} + \frac{1}{n}}} \qquad \textbf{(EQ 37)}$$

is a standard *t* variable (i.e., with zero mean and unit variance) with *m+n-2* degrees of freedom. Therefore, we can use a *t* test to determine whether *c* has a zero mean, using the approach in 2.5.4 on page 44.

**Example 13: Comparing two samples when sample sizes differ**

Continuing with Example 12, assume that we have additional data points for the simulation runs with 5 buffers, as shown below. Can we still claim that the buffer size plays a role in determining the loss rate?

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Loss rate with 5 buffers** | 0.20% | 0.30% | 0.90% | 1.40% | 1.00% | 0.80% | 1.10% | 0.20% | 1.50% | 0.20% | 0.50% | 1.20% | 0.70% | 1.30% | 0.90% |
| **Loss rate with 100 buffers** | 0.10% | 0.60% | 1.10% | 0.80% | 1.20% | 0.30% | 0.70% | 1.90% | 0.20% | 1.20% | | | | | |

Here, *m* = 15 and *n* = 10, so we cannot use the approach of Example 12. Instead, we will first compute the mean and confidence intervals of both samples to see if the intervals overlap. It is easily found that, at the 95% level, using a *t* distribution with 14 and 9 degrees of freedom respectively, the confidence intervals are 0.81±0.25 and 0.81±0.40 which overlap. However, the sample variances are not the same, and there is a good chance that the population variances also differ. Nevertheless, for the purpose of this example, we will make the assumption that the population variances are the same. Therefore, we compute $s^2$ using Equation 36 as 2.89, so that *s* = 1.7. We then compute *c* as $0.0033/(1.7(1/15 + 1/10)^{1/2}) = 0.0048$. Since this has unit variance, it is easy to see using the *t* test with 23 degrees of freedom that 0 lies in the confidence interval for *c*, which implies that, with this data set, buffer size has no statistically significant effect on packet loss rate.

[]

(2) *Sample variances differ, but* m *and* n *are both large.* In this case, it can be shown that the variable *c* defined by:

$$c = \frac{\bar{a} - \bar{b}}{\sqrt{\left(\frac{\sum_{i=1}^{n} (a_i - \bar{a})^2}{m(m-1)}\right) + \left(\frac{\sum_{i=1}^{m} (b_i - \bar{b})^2}{n(n-1)}\right)}} \quad \text{(EQ 38)}$$

is a standard normal variable (i.e., with a zero mean and unit variance). Therefore, we can use a standard normal test to determine whether *c* has a zero mean, using the approach in 2.5.4 on page 44.

If neither assumption can be made, then it is difficult to draw meaningful comparisons, other than by using *non-parametric tests*, in this case the Mann-Whitney U test, which is beyond the scope of this text. Therefore, we will not consider this case further.

### 2.5.6 Testing hypotheses regarding quantities measured on ordinal scales

So far, we have tested hypotheses where a variable takes on real values. We now consider the case where a variable takes on nominal values (such as ‘UDP’ or ‘TCP’) or ordinal values (such as ‘bad’, ‘satisfactory’, and ‘good’) (these terms are defined in 2.2.2 on page 32). In such cases, hypothesis testing using the techniques described above is meaningless because a sample cannot be described by a mean, nor can we define real confidence intervals about the mean. Instead, for such variables, hypotheses are of the form:

*$H_0$: the observed values are drawn from an expected distribution*

Then, we use a statistical test, such as the *Pearson chi-squared test* (described below), to reject or not reject the hypothesis.

**Example 14: Hypothesis formulation with nominal scales**

Suppose that you want to check if the distribution of packet types on a link from your campus to the Internet is similar to that reported in the literature. For instance, according to [Genevieve Bartlett, John Heidemann, Christos Papadopoulos, and James Pepin. Estimating P2P Traffic Volume at USC. Technical Report ISI-TR-2007-645, USC/Information Sciences Institute, June, 2007], 42% of academic traffic by bytes at the University of Southern California (USC) can be attributed to P2P applications. Suppose you measure 100 GB of traffic and find that 38 GB are due to P2P applications. Then, a reasonable null hypothesis would be:

*$H_0$: the observed traffic on the campus Internet access link is similar to that at USC*

[]

How should we test hypotheses of this form? A clue comes from the following thought experiment. Suppose we have a possibly biased coin and that we want to determine whether it is biased or not. The null hypothesis is:

*$H_0$: P(heads) = P(tails) = 0.5*

We assume that we can toss the coin as many times as we want and that the outcome of each toss is independent. Let T denote the outcome ‘Tails’ and H denote the outcome ‘Heads.’ We will represent a set of outcomes such as ‘Nine heads and one tails’ by the notation $TH^9$. As we saw earlier, if a coin is unbiased, this outcome has the probability $\binom{10}{1}0.5^9 0.5^1$. Any outcome from *n* coin tosses—such as *a* Heads, represented by $H^aT^{n-a}$–can be viewed as one sample drawn at random from the set of all possible outcomes when tossing a coin *n* times. A little thought indicates that the probability of this outcome, given that the probability of heads is $p$ and of tails is $q=1-p$, is given by $\binom{n}{a}p^a q^{n-a}$, which is also the $a$th term of the expansion of the expression $(p + q)^n$. As $n \to \infty$, the binomial distribution tends to the normal distribution, so that the probability of each outcome is approximated by the normal distribution.

Now, consider an experiment where each individual outcome is independent of the others, and where an outcome results in one of $k$ ordinal values, $o_1, o_2, \dots, o_k$. Let the expected probability of the $i$th outcome be $p_i$, so that the expected count for the $i$th outcome, $e_i=np_i$. Suppose we run the experiment $n$ times, and the $i$th outcome occurs $n_i$ times with $\sum_i n_i = n$. We can represent any particular outcome by $o_1^{n_1}o_2^{n_2}\dots o_k^{n_k}$ and this outcome can be viewed as one sample drawn at random from the set of all possible outcomes. The probability of such an outcome is given by the *multinomial* distribution as:

$$P\left(o_1^{a_1}o_2^{a_2}\dots o_k^{a_k}\right) = \binom{n}{n_1}\binom{n-n_1}{n_2}\dots\binom{n-\sum_i^{k-1} n_i}{n_k}p_1^{n_1}p_2^{n_2}\dots p_k^{n_k} \qquad \textbf{(EQ 39)}$$

$$= \frac{n!}{n_1!n_2!\dots n_k!}p_1^{n_1}p_2^{n_2}\dots p_k^{n_k} \qquad \textbf{(EQ 40)}$$

This outcome is one of the terms from the expansion of $(p_1+p_2+\dots+p_k)^n$. As with the binomial distribution, we can use the multinomial distribution to test if any particular outcome, conditional on a null hypothesis on the $p_i$s being true, is ‘too unlikely,’ indicating that the null hypothesis should be rejected.

In many cases, using the multinomial distribution for testing the validity of a hypothesis can be cumbersome. Instead, we use a standard mathematical result that the variable $X_i = \frac{n_i - e_i}{\sqrt{e_i}}$, for values of $e_i > 5$, closely approximates a standard normal variable with zero mean and unit variance. But we immediately run into a snag: the $n_i$ are not independent. For example, if $n_3 = n$, then all the other $n_i$ must be zero. Therefore, the $X_i$ are also not independent. However, it can be proved that a set of $k$ *dependent* variables $X_i$ can be mapped through an orthogonal transformation to a set of $k$-1 *independent* standard normal variables while keeping the sums of squares of the variables constant. By definition, the sum of squares of $k$-1 independent standard normal variables follows the $\chi^2$ (written chi-squared and pronounced kai-squared) distribution with $k$-1 degrees of freedom. Therefore, if the null hypothesis is true (that is, the observed quantities are drawn from the distribution specified implicitly by the expected values) the variable

$$X = \sum_{i=1}^{k}\frac{(n_i - e_i)^2}{e_i} \qquad \textbf{(EQ 41)}$$

is a $\chi^2$ variable with $k$-1 degrees of freedom. Standard statistical tables tabulate $P(X > a)$ where $X$ is a $\chi^2$ variable with $k$ degrees of freedom. We can use this table to compute the degree to which a set of observations corresponds to a set of expected values for these observations. This test is the *Pearson $\chi^2$ test.*

**Example 15: (Chi-squared test)**

We use the Pearson $\chi^2$ test to test if the observation in Example 14 results in rejection of the null hypothesis. Denote P2P traffic by ordinal 1 and non-P2P traffic by ordinal 2. Then, $e_1 = 42$, $e_2 = 58$, $n_1 = 38$, $n_2 = 62$. Therefore, $X = (38\text{-}42)^2/42 + (62\text{-}58)^2/58 = 0.65$. From the $\chi^2$ table with 1 degree of freedom, we see that $P(X > 3.84) = 0.05$, so that any value greater than 3.84 occurs with probability less than 95% and is ‘unlikely.’ Since 0.65 < 3.84, the observation is not unlikely, which means that we cannot reject the null hypothesis.

In contrast, suppose the observation was $n_1 = 72$, $n_2 = 28$. Then, $X = (72\text{-}42)^2/42 + (28\text{-}58)^2/58 = 36.9$. Since 36.9 > 3.84, such an observation would suggest that we should reject the null hypothesis at the 5% level.

[]

### 2.5.7 Fitting a distribution

When testing a hypothesis using a chi-square test we need to compute the expected distribution of sample values. These expected values may come from prior studies, as in the example above, or from physical considerations. In many cases, however, the expected values can be derived by assuming that the observations arise from a standard distribution (such as the Poisson, exponential, or normal distributions) and then choosing the parameters of the distribution to best match the observed values. This is called 'fitting' a distribution to the observations. A general technique for fitting a distribution is called the *method of maximum likelihood* and we discuss it next.

Suppose that random variables $X_1, X_2,...,X_n$ have a (known) joint density function $f_\theta(x_1,x_2,...,x_n)$ where $\theta$ denotes the unknown parameters of the distribution, such as its mean and variance. Given the observation $X_i=x_i$, where $i$=1,2,...,$n$, we would like to compute the *maximum likelihood estimate (mle) of* $\theta$, that is, the value of $\theta$ that makes the observed data the 'most likely.' Intuitively, conditional on the observations being what they are, we would like to work backwards to find the value of $\theta$ that made these observations likely: we then assume that we observed what we did because the parameters were what they were.

Assuming that the $X_i$s are independent and identically distributed according to $f_\theta(.)$, the joint probability that the observation is $(x_1,x_2,...,x_n)$ is simply the product of the individual probabilities $\prod_{i=1}^{n} f_\theta(X_i)$. Note that the distribution function is parametrized by $\theta$. We make this explicit by defining *likelihood(*$\theta$*)* as

$$likelihood(\theta|x_1,x_2,\ldots,x_n) = \prod_{i=1}^{n} f_\theta(X_i) \quad \textbf{(EQ 42)}$$

We find the mle by maximizing *likelihood(*$\theta$*)* with respect to $\theta$. In practice, it is more convenient to maximize the natural logarithm of *likelihood(.)* denoted *l(.),* defined by

$$l(\theta|x_1,x_2,\ldots,x_n) = \sum_{i=1}^{n} \log(f_\theta(X_i)) \quad \textbf{(EQ 43)}$$

For example, suppose that we want to fit a Poisson distribution with parameter $\lambda$ to an observation $(x_1,x_2,...,x_n)$. Recall that for a Poisson distribution, $P(X=x) = \frac{\lambda^x e^{-\lambda}}{x!}$. If the $X$ are independent and identically distributed (i.i.d.) Poisson variables, their joint probability is the product of their individual distributions, so that

$$\begin{aligned} l(\lambda) &= \sum_{i=1}^{n} (X_i \log\lambda - \lambda - \log X_i!) \\ l(\lambda) &= \log\lambda \sum_{i=1}^{n} X_i - n\lambda - \sum_{i=1}^{n} \log X_i! \end{aligned} \quad \textbf{(EQ 44)}$$

We maximize *l(.)* by differentiating it with respect to $\lambda$ and setting the derivative to 0:

$$\frac{dl}{d\lambda} = \frac{1}{\lambda}\sum_{i=1}^{n} X_i - n = 0 \quad \textbf{(EQ 45)}$$

which yields the satisfying result:

$$\lambda = \bar{X} \qquad \text{(EQ 46)}$$

Thus, we have found that the mean of a set of observations is the value that maximizes the probability that we obtain that particular set of observations, conditional on the observations being independent and identically distributed Poisson variables.

Proceeding along similar lines, it is possible to show that the maximum likelihood estimators for a set of i.i.d normal variables is

$$\mu = \bar{X}$$
$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(X_i - \bar{X})^2} \qquad \text{(EQ 47)}$$

Note the mle for the standard deviation is not a consistent estimator - to get a consistent estimator, we need to divide by *n*-1, rather than *n*, as discussed in 2.3.5 on page 37. Maximum likelihood estimators for other distributions can be found in standard texts on mathematical statistics.

It is possible to obtain confidence intervals for maximum likelihood estimators by considering the sampling distribution of the estimated parameters. This is discussed in greater depth in more advanced texts, such as (J.A. Rice, Mathematical Statistics and Data Analysis, 3e, Thomson, 2007. Section 8.5.3.).

Note that if we use the sample itself to estimate *p* parameter values of the population, then we reduce the number of degrees of freedom in the sample by *p*. Recall that a sample that has *n* counts (ordinal types), has *n-1* degrees of freedom. If, in addition, *p* parameters are estimated to compute the expected counts, then the degrees of freedom when conducting a chi-squared test is *n-1-p.*

**Example 16: (Fitting a Poisson distribution)**

In an experiment, a researcher counted the number of packet arriving to a switch in each 1ms time period. The table below shows the count of the number of time periods with a certain number of packet arrivals. For instance, there were 146 time periods that had 6 arrivals. The researcher expects the packet arrival process to be a Poisson process. Find the best Poisson fit for the sample. Use this to compute the expected count for each number of arrivals. What is the chi-squared variable value for this data set? Determine whether the Poisson distribution adequately describes the data.

| Number of packet arrivals | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Count | 18 | 28 | 56 | 105 | 126 | 146 | 164 | 165 | 120 | 103 | 73 | 54 | 23 | 16 | 9 | 5 |

The total number of time periods is 18+28+...+5 = 1211. The total number of arrivals is (18*1)+(28*2)+...+(5*16) = 8935. Therefore, the mean number of packets arriving in 1ms is 8935/1211 = 7.38. This is the best estimate for the mean of a fitted Poisson distribution. We use this to generate the probability of a certain number of arrivals in each 1ms time period. This probability multiplied by the total number of time periods is the expected count for that number of arrivals, and this is shown below. For instance, we compute P(1) = 0.0046 and 0.0046*1211 = 6.

| Number of packet arrivals | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Count | 18 | 28 | 56 | 105 | 126 | 146 | 164 | 165 | 120 | 103 | 73 | 54 | 23 | 16 | 9 | 5 |
| Expected count | 6 | 21 | 51 | 93 | 138 | 170 | 179 | 165 | 135 | 100 | 67 | 41 | 23 | 12 | 6 | 3 |

Although the fit papers to be a good one from visual inspection, it is best to compute the chi-squared value. This is computed as $(18-6)^2/6) + (28-21)^2/21 +.... + (5-3)^2/3 = 48.5$. Since we estimated one parameter from the sample, the degrees of freedom = 16-1-1 = 14. From the chi-squared table, with 14 degrees of freedom, at the 95% confidence level, the critical value is 23.68. Therefore, we reject the hypothesis that the sample is well-described by a Poisson distribution at this confidence level. That is, we have 95% confidence that this sample was not drawn from a Poisson population. The critical value at the 99.9% level for 14 degrees of freedom is 36.12. So, we can be even stronger, and state that with 99.9% confidence, the sample is not drawn from a Poisson population.

At first glance, this is a surprising result, because the fit appears quite good. The reason why the test fails is clear when we examine the $(O-E)^2/E$ values. The largest value is 27.6, which is for 1 packet arrival, where we expected a count of 6 but got 18. Because the denominator here is small (6), the contribution of this sample value to the chi-squared variable is disproportionate. If we were to ignore this value as an outlier and computed the fit only for 2-16 packet arrivals, then the revised estimate of the distribution mean is 7.47, and the revised chi-squared variable is 19.98 (see Exercise 12). This does meet the goodness-of-fit criterion with 13 degrees of freedom even at the 95% confidence level. In cases like these, it is worthwhile looking into why there was a deviation from the Poisson process: it could be a systematic error in the experiment, or perhaps due to a heretofore unknown phenomenon.

[]

### 2.5.8 Power

Recall that when we test a hypothesis, we determine the probability of obtaining an observed outcome conditional on the null hypothesis being true. If the outcome is less probable than the *significance level* such as 95% or 99%, then we reject the null hypothesis. Of course, the hypothesis could still be true. Nevertheless, we reduce the Type I error, that of rejecting a hypothesis when it is in fact true, to a value below the significance level.

We now discuss a related concept: the power of a test. The power of a statistical test is the probability that the test will reject a null hypothesis when it is in fact false. If the power is low, then we may not reject a null hypothesis even when it is false, a Type II error. Thus, the greater the power, the lower the chance of making a Type II error. Usually, the only way to increase the power of a test is to increase its significance level (which makes a Type I error more likely).

The practical difficulty in computing the power of a test is that we don’t know the ground truth. So, it becomes impossible to compute the probability that we will reject the null hypothesis conditional on the ground truth being different from the null hypothesis. For instance, suppose that the ground truth differs infinitesimally from the null hypothesis. Then, the probability that we reject the null hypothesis (which is false) is essentially the same as the significance level (why?). On the other hand, suppose that the ground truth is far from the null hypothesis. Then, the sample mean is likely to be near the ground truth and we are likely to reject the null hypothesis, increasing the power of the test. But we have no way of knowing which of these situations hold. Therefore, we can only precisely compute the power of a test in the context of an alternative hypothesis about the state of the world. Unfortunately, in many cases, this is impossible to determine. Therefore, despite its intuitive merit, the power of a test is rarely computed.

## *2.6 Independence and dependence: regression, and correlation*

Thus far, we have, for the most part, studied single variables in isolation. In this section, we study data sets with two variables. In this situation, some questions immediately crop up: Are the variables independent of each other? If not, are pairs of variables correlated with each other? Do some of the variables depend linearly or non-linearly on the others? Can the variability in one of the variables be explained as being due to variability in another variable? These are the types of questions that we will study in this section.

### 2.6.1 Independence

Consider a data set where each element can be simultaneously placed into more than one category. For example, we could characterize a an IP packet both by its size and its type. Are these variables independent of each other? In other words, given the size, can we say anything about the type and vice versa? If knowing the value of one variable does not give us any additional information about the other, then the variables are *independent*. We now describe a test to determine whether we can confidently reject the hypothesis that two variables are independent.

In testing for independence, it is useful to represent the data set in the form of a *contingency table*. For a sample that has two variables that take one of *m* and *n* ordinal values respectively, the contingency table has *m*x*n* cells, with each cell containing the count of the number of sample elements that simultaneously fall into both corresponding categories.

Given the contingency table, we can use the Pearson chi-squared test to test whether two variables are independent as follows. We use the sample to estimate the population parameters, and, from these estimates, assuming independence of the variables, we compute the expected numbers of sample values that will fall into each cell of the contingency table. We can then compare the actual counts in each cell with these expected values to compute the chi-squared statistic. If this statistic is larger than the critical value, then we can, with high confidence, reject the hypothesis that the variables are independent.

In computing the chi-squared statistic, it is important to correctly compute the degrees of freedom. Recall that for a variable that falls into one of *k* classes, the number of degrees of freedom is *k*-1. It can be shown that the number of degrees of freedom is further reduced by each parameter that is estimated from the sample (instead of being known *a priori*) as the next example shows.

**Example 17: (Testing for independence)**

Suppose that in a packet trace with a million packets, you obtain the following contingency table:

| | **TCP** | **UDP** | **Other** | **Row sum** |
|---|---|---|---|---|
| **40** | 12412 | 15465 | 300 | **28177** |
| **100** | 85646 | 12561 | 15613 | **113820** |
| **150** | 9846 | 68463 | 4561 | **82870** |
| **512** | 4865 | 45646 | 23168 | **73679** |
| **1024** | 48651 | 95965 | 48913 | **193529** |
| **1200** | 98419 | 59678 | 48964 | **207061** |
| **1450** | 156461 | 48916 | 51952 | **257329** |
| **1500** | 16516 | 24943 | 2076 | **43535** |
| **Column sum** | **432816** | **371637** | **195547** | **1000000** |

Is the packet size independent of the packet type?

We do not know the actual frequency of each packet type in the population (of all IP packets), so we will estimate the population frequencies from this sample using the column sums as follows:

P(TCP) = 432816/1000000 = 0.433

P(UDP) = 371637/1000000 = 0.372

P(Other) = 195547/1000000 = 0.195

Similarly, we compute the probability of each packet size from the row sums as follows:

P(40) = 28177/100000 = 0.028

P(100) = 113820/1000000 = 0.114

...

P(1500) = 43535/1000000 = 0.043

If these probabilities were independent, then each cell could be computed as follows:

Count of TCP AND 40 = P(TCP) * P(40) * 1000000 = 0.433*0.028*1000000 = 12195.

...

Count of Other and 1500 = P(Other) * P(1500) * 1000000 =0.195* 0.043*1000000 = 8513.

We therefore have both the observed and expected values for each cell. We compute the chi-squared statistic as the sum of squares of the variable (observed value - expected value)$^2$/(expected value). This value turns out to be 254,326. Here *k* is 3*8 = 24. Moreover, we have estimated nine parameters from the data (we get two probabilities 'for free' since probabilities sum to 1). Therefore, the degrees of freedom still left is 24-1-9 = 14. Looking up the chi-square table with 14 degrees of freedom, we find that the critical value for the 0.001 confidence level to be 36.12. Since the statistic far exceeds this value, we can be more than 99.9% confident that packet type and packet size are *not* independent in the population from which this trace was drawn.

[]

Note that, given the large sample size, even a tiny deviation from the expected values will lead to the null hypothesis of independence to be rejected. We discuss this further in 2.10.5 on page 63.

### 2.6.2 Regression

When two random variables are not independent, it is sometimes the case that one variable depends on—or can be thought to depend on—the other, in that the value of the second variable is approximately known if the value of the first is known. Let the independent variable *X* take on specific values such as *x*, and let the dependent variable be *Y*. Then, the *regression* of *Y* on *x* is a graph that shows *E(Y|x)* as a function of *x*. Note that this graph is only defined when both *X* and *Y* are defined on interval or ratio scales (why?).

In the simplest case, we model *Y* as varying linearly with *x*. In this case, we define a *best-fit line* that minimizes the sum of squared deviations of observed from the estimated values of *y*. If our observations are of the form $\{(x_i,y_i)\}$, then the model for the regression line is:

$$y = a + bx \qquad \text{(EQ 48)}$$

and therefore we seek to minimize

$$S^2 = \sum_{i=1}^{n} (y_i - a - bx_i)^2 \qquad \text{(EQ 49)}$$

To find *a* and *b* we set the partial derivative of *S* with respect to *a* and *b* to zero. This gives us the

$$a = \bar{y} - b\bar{x}$$
$$b = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})^2} \qquad \text{(EQ 50)}$$

Substituting for *a* in Equation 48 we see that the point $(\bar{x}, \bar{y})$ satisfies the regression equation, so that the regression line always passes through this point, which is also called the *centroid* of the sample. We interpret *a* as the Y intercept of the best-fit line. *b* is the mean change in *Y* with a unit increase in *X*.

When *Y* does not depend on *X* linearly, it is sometimes possible to transform *Y* so that the dependency is more nearly linear, as the next example demonstrates.

**Example 18: (Computing a linear regression after transformation)**

Consider the following data set, which shows the packet loss rate for a given buffer size, where three simulations were run for each buffer size setting. Compute the linear regression of the loss rate on the buffer size.

| Buffer size | 10 packets | 20 packets | 50 packets | 100 packets | 200 packets | 500 packets |
|---|---|---|---|---|---|---|
| **Run 1** | 30.20 | 10.20 | 5.20 | 1.10 | 0.20 | 0.01 |
| **Run 2** | 27.40 | 11.30 | 6.37 | 1.70 | 0.23 | 0.01 |
| **Run 3** | 29.10 | 9.80 | 5.82 | 1.30 | 0.17 | 0.01 |

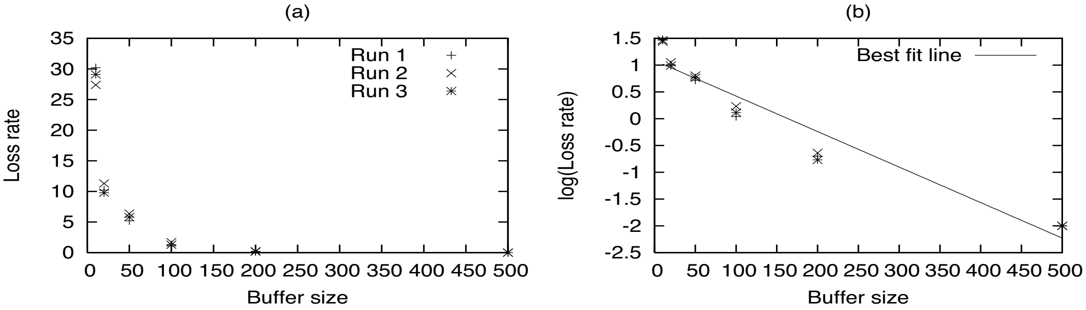


**FIGURE 5.** **Scatter plot of loss rate vs. buffer size**

It is instructive to look at the scatter plot of the data, shown in Figure 5(a). It is immediately obvious that the relationship between the loss rate and the buffer size is far from linear. In such cases, it is necessary to transform the data values to extract a more linear relationship. Figure 5(b) is a scatter plot which plots the logarithm of the loss rate with respect to the buffer size. It is clear that the relationship is far more linear than before. We compute the best-fit line, using Equation 50, as $y = 1.0568 - 0.0066\,x$ which is the regression of $Y$ on $x$. This best fit line is also shown in the figure.

[]

The best-fit line, in addition to minimizing the sum of squared errors, has several useful properties. It is the maximum likelihood estimator for the population parameters *a* and *b*. Moreover, it is possible to construct confidence intervals for the values of *a* and *b* by means of the *t* distribution.

Note that it is always possible to compute a linear regression of one variable on another, even if the two variables are not linearly related. Therefore, it is always a good idea to use a statistical test for linearity (or the correlation coefficient, described next), to validate that the relationship is reasonably linear before computing the best-fit line.

We now briefly discuss three extensions to simple linear regression:

- The least-squares best-fit approach assumes that the degree of variation in the dependent variable is more or less the same, regardless of the value of the independent variable. In some cases, the greater (or smaller) the value of the independent variable, the greater the variance in the dependent variable. For example, in the preceding example, we see a greater variation in log(loss rate) for smaller values of the buffer size. Such samples are said to be *heteroscedastic*, and if the departure from uniform variability is significant, it is necessary to resort to advanced techniques to compute the regression.
- In some cases, the relationship between the dependent and independent variable is non-linear even after transformation of the dependent values. In such cases, it may become necessary to perform *non-linear* regression.

- Finally, we have considered a dependent variable that depends on only a single independent variable. In general, the dependency may extend to multiple independent variables. This is the subject of *multiple regression*.

These three topics are treated in greater depth in more advanced texts on statistics, such as (Snedecor, Rice).

### 2.6.3 Correlation

When computing a regression, we can use physical considerations to clearly identify independent and dependent variables. In some cases, however, the outcomes of an experiment can be thought of as being mutually dependent on each other. This dependency is captured in the statistical concept of *correlation*. Moreover, as we will see later, even if one variable depends on the other, the correlation coefficient allows us to determine the degree to which variations in the dependent variable can be explained as a consequence of variations in the independent variable.

**Example 19: (Correlated variables)**

Suppose we transfer a small file over a cellular modem ten times, each time measuring the *round-trip delay* (from a 'ping' done just before transferring the file) and the *throughput* achieved (by dividing the file size by the transfer time). The round-trip delay may be large because the network interface card may have a low capacity, so that even a small ping packet experiences significant delays. On the other hand, the file transfer throughput may be low because the path delay is large. So, it is not clear which variable ought to be the dependent variable and which variable ought to be the independent variable. Suppose that the measured round-trip delays and throughputs are as shown below:

| **Throughput (kbps)** | 46 | 65 | 53 | 38 | 61 | 89 | 59 | 60 | 73 |
|---|---|---|---|---|---|---|---|---|---|
| **Round-trip delay (ms)** | 940 | 790 | 910 | 1020 | 540 | 340 | 810 | 720 | 830 |

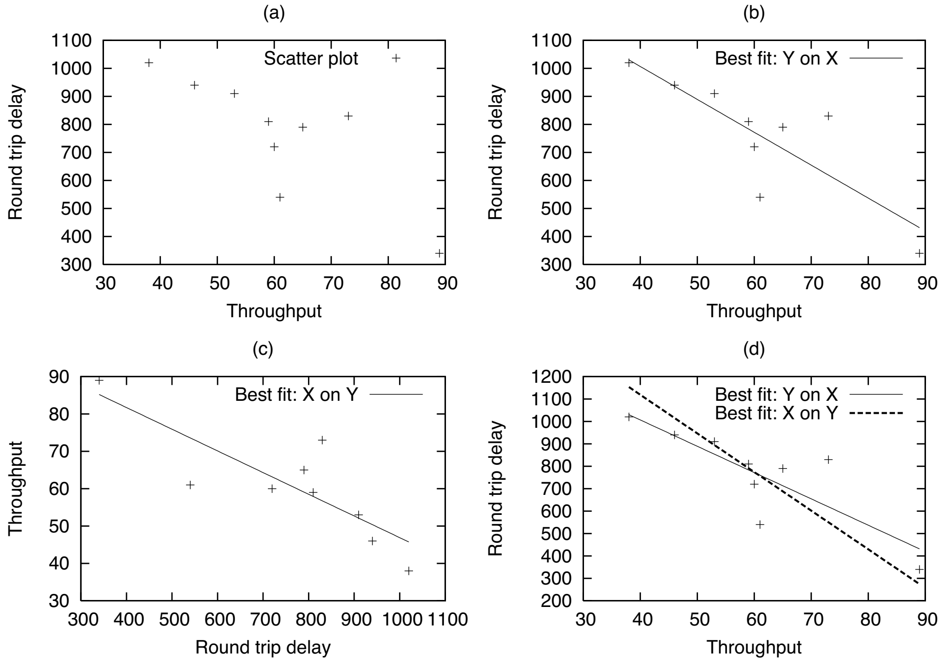


**FIGURE 6. Regression and correlation**

Figure 6(a) shows the scatter plot of the two variables. There appears to be an approximately linear decline in the round-trip delay with an increase in throughput. We arbitrarily choose throughput to be the independent variable and do a regression of round-trip delay on it, as shown in Figure 6(b). We see that the best-fit line has a negative slope, as expected.

Now, there is no reason why we could not have chosen the round-trip delay to be the independent variable and have done a similar regression. This is shown in Figure 6(c). Again, we see that as the round-trip delay increases, the throughput decreases, indicating a negative relationship. We also see the best-fit line with a negative slope.

Note that the two regression lines are not the same! In one case, we are trying to minimize the sum of the squared errors in round-trip delay, and in the other, the sum of squared errors in throughput. So, the best fit lines will, in general, not be the same. This is shown in Figure 6(d) where we show both best-fit lines (one drawn with transposed axes).

[]

The reason why two regression lines do not coincide in general is best understood by doing a thought experiment. Suppose two outcomes of an experiment, say *X* and *Y*, are completely independent. Then, *E(XY) = E(X)E(Y)*, by definition of independence. In the context of a single sample, we rewrite this as:

$$E(XY) = \frac{\sum x_i y_i}{n} = E(X)E(Y) = \frac{\sum x_i \sum y_i}{n \quad n} = \bar{x}\bar{y} \qquad \textbf{(EQ 51)}$$

Recall that $b = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})^2}$. We expand the numerator as $\sum x_i y_i - \bar{x}\sum y_i - \bar{y}\sum x_i + n\bar{x}\bar{y}$. Rewriting $\sum y_i$ as $n\bar{y}$ and $\sum x_i$ as $n\bar{x}$, and using Equation 51, we get

$$b = \frac{\sum x_i y_i - \bar{x}\sum y_i - \bar{y}\sum x_i + n\bar{x}\bar{y}}{\sum (x_i - \bar{x})^2} = \frac{n\bar{x}\bar{y} - n\bar{x}\bar{y} - n\bar{x}\bar{y} + n\bar{x}\bar{y}}{\sum (x_i - \bar{x})^2} = 0 \qquad \textbf{(EQ 52)}$$

so that the regression line has zero slope, i.e., is parallel to the X axis. Symmetrically, the regression of X on Y will be parallel to the Y axis (Why?). Therefore, the two regression lines meet at right angles when the outcomes are independent. Recalling the we can interpret *b* as the expected increment in *Y* with a unit change in *X*, *b* = 0 implies that a unit change in *X* does not change *Y* (in expectation), which is consistent with independence.

On the other hand, if one outcome is perfectly linearly related to the other, then *Y* = *tX*. Clearly, $\bar{y} = t\bar{x}$, so that

$b = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})^2} = \frac{\sum (x_i - \bar{x})(tx_i - t\bar{x})}{\sum (x_i - \bar{x})^2} = t$. Denoting the regression of X on Y by *x = a' + b'y,* the expression for *b'* is given by $\frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (y_i - \bar{y})^2} = \frac{\sum (x_i - \bar{x})(tx_i - t\bar{x})}{\sum (tx_i - t\bar{x})^2} = \frac{1}{t}$. With transposed axes, this line exactly overlaps the best fit line for the regression of Y on X. In other words, when there is exact linear dependence between the variables, the best fit regression lines meet at zero degrees. Thus, we can use the angle between the regression lines as an indication of the degree of linear dependence between the variables.

In practice, the standard measure of dependence, or *correlation*, is the square root of the product *bb'*, denoted *r*, also called *Pearson's correlation coefficient*, and is given by

$$r = \sqrt{\frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})^2}\frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (y_i - \bar{y})^2}} = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\left(\sum (x_i - \bar{x})^2\right)\left(\sum (y_i - \bar{y})^2\right)}} \qquad \textbf{(EQ 53)}$$

When the slopes are perpendicular, $r = 0$, and when the slopes are inverses of each other, so that the regression lines overlap, then $r = 1$. Moreover, when $X$ and $Y$ are perfectly negatively correlated, so that $Y = -tX$, $r = -1$ (Why?). Therefore, we interpret $r$ as the degree of correlation between two variables, ranging from -1 to +1, with its sign indicating direction of correlation (positive or negative), and its magnitude indicating the degree of correlation.

**Example 21: Correlation coefficient**

Compute the correlation coefficient for the variables in Example 17.

We compute the mean throughput as 54.4 kbps and the mean delay as 690 ms. Substituting these values into Equation 53, we find that $r = -0.56$. This indicates a negative correlation, but it is not particularly linear.

[]

There are many interpretations of the correlation coefficient (See Joseph Lee Rodgers and W. Alan Nicewander, “Thirteen Ways to Look at the Correlation Coefficient,” *The American Statistician*, Vol. 42, No. 1 (Feb., 1988), pp. 59-66). One particularly insightful interpretation is based on the sum of squares minimized in a linear regression: $S^2 = \sum_{i=1}^{n}(y_i - a - bx_i)^2$. Substituting for $a$ and $b$, it is easily shown (see Exercise 14) that

$$S^2 = (1 - r^2)\sum(y_i - \bar{y})^2 \qquad \textbf{(EQ 54)}$$

That is, $r^2$ is the degree to which a regression is able to reduce the sum of squared errors, which we interpret as the degree to which the independent variable explains variations in the dependent variable. When we have perfect linear dependency between $Y$ and $X$, then the degree of correlation is 1 in absolute value, and the regression line is perfectly aligned with the data, so that it has zero error.

In computing a correlation coefficient, it is important to remember that it only captures linear dependence. A coefficent of zero does not mean that the variables are independent: they could well be non-linearly dependent. For example, if $y^2 = 1 - x^2$, then for every value of $X$, there are two equal and opposite values of $Y$, so that the best fit regression line is the X axis, which leads to a correlation coefficient of 0. But, of course, $Y$ is not independent of $X$! Therefore, it is important to be cautious in drawing conclusions regarding independence when using the correlation coefficient. For drawing such conclusions, it is best to use the chi-square goodness-of-fit test described earlier.

Like any statistic, the correlation coefficient $r$ can have an error due to random fluctuations in a sample. It can be shown that if $X$ and $Y$ are jointly normally distributed, then the variable $z = 0.5\log\left(\frac{1+r}{1-r}\right)$ is approximately normally distributed with a mean of $0.5\ \log\left(\frac{1+\rho}{1-\rho}\right)$ and a variance of $1/(n-3)$. This can be used to find the confidence interval around $r$ in which we can expect to find $\rho$.

A specific form of correlation that is relevant in the analysis of time series is *autocorrelation*. Consider a series of values of a random variable that are indexed by discrete time, i.e., $X_1, X_2, ..., X_n$. Then, the autocorrelation of this series with lag $l$ is the correlation coefficient between the random variable $X_i$ and $X_{i-l}$. If this coefficient is large (close to 1) for a certain value of $l$, then we can infer that the series has variation on the time scale of $l$. This is often much easier to compute than a full scale harmonic analysis by means of a Fourier transform.

Finally, it is important to recognize that correlation is not the same as causality. We must not interpret a correlation coefficient close to 1 or -1 to infer causality. For example, it may be the case that packet losses on a wireless network are positively correlated with mean frame size. One cannot infer that larger frame sizes are more likely to be dropped. It could be the case, for example, that the network is heavily loaded when it is subjected to video traffic, which uses large frames. The increase in the loss rate could be due to the load, rather than the frame size. Yet, the correlation between these two quantities would be strong.

To go from correlation to causation, it is necessary to determine the physical causes that lead to causation. Otherwise, the unwary researcher may be led to unsupportable and erroneous conclusions.

## *2.7 Comparing multiple outcomes simultaneously: analysis of variance*

In the discussion so far, we have focused on determining dependence and independence between two variables (2.6 on page 51) and comparing the outcomes of experiments corresponding to at most two choices of experimental controls, or 'treatments' (2.5.5 on page 45). Suppose we wanted to compare outcomes of multiple treatments simultaneously. For example, Examples 12 and 13 compared the packet loss rate at a queue with 5 buffers with the loss rate at a queue with 100 buffers. Instead, we may want to compare the loss rates with 5, 100, 200, 500, and 1000 buffers with each other. How should we proceed?

Theoretically, we could perform a set of pairwise comparisons, where each comparison used a normal or t test. For example, we could test the hypothesis that the loss rates with 5 and 100 buffers were identical, the hypothesis that the loss rates with 5 and 200 buffers were identical, and so on. This approach, however, leads to a subtle problem. Recall that when we reject a hypothesis, we are subject to a Type I error, that is, rejecting a hypothesis that is true. If we perform many pairwise comparisons, although the probability of Type I error for any one test is guaranteed to be below 5% (or 1%), the overall probability of making at least one Type I error can be greater than 5% (or 1%)! To see this, think of flipping a coin ten times and looking for ten heads. This has a probability of about 1/1024 = 0.1%. But if we were to flip 1024 coins, chances are good that we would get at least one run of ten heads. Arguing along similar lines it is easy to see that, as the number of comparisons increases, the overall possibility of a Type I error increases. What is needed, therefore, is a way to perform a single test that avoids numerous pairwise comparisons. This is achieved by the technique of 'Analysis of Variance' or ANOVA.

ANOVA is a complex topic with considerable depth. We will only discuss the simplest case of the 'one-way layout' with fixed effects. Multi-way testing and random effects are discussed in greater depth in texts such as (John A. Rice, "Mathematical Statistics and Data Analysis, 3e, Thomson Books, 2007).

### 2.7.1 One-way layout

In the analysis of a one-way layout, we group observations according to their corresponding treatment. For instance, we group repeated measurements of the packet loss rate for a given buffer size, say 5 buffers. The key idea in ANOVA is that if none of the treatments–such as the buffer size– affect the observed variable–such as the loss rate–then all the observations would be drawn from the same population. Therefore, the sample mean computed for observations corresponding to each treatment should not be too far from the sample mean computed across all the observations. Moreover, the estimate of population variance computed from each group separately should not differ too much from the variance estimated from the entire sample. If we do find a significant difference between statistics computed from each group separately and the sample as a whole, we reject the null hypothesis. That is, we conclude that, with high probability, the treatments affect the observed outcomes. By itself, that is all that basic ANOVA can tell us. Further testing is necessary to determine which treatments affect the outcome and which do not.

We now make this more precise. Suppose we can divide the observations into *I* groups of *J* samples each (we assume that all groups have the same number of samples: this is usually not a problem because the treatments are under the control of the experimenter). We denote the *j*th observation of the *i*th treatment by the random variable $Y_{ij}$. We model this observation as the sum of an underlying population mean $\mu$, the true effect of the *i*th treatment $\alpha_i$, and a random fluctuation $\varepsilon_{ij}$:

$$Y_{ij} = \mu + \alpha_i + \varepsilon_{ij} \tag{EQ 55}$$

These errors are assumed to be independent and normally distributed with zero mean and a variance of $\sigma^2$. For convenience, we normalize the $\alpha_i$s so that $\sum_{i=1}^{I} \alpha_i = 0$. Note that the expected outcome for the *i*th treatment is $E(Y_{ij}) = \mu + \alpha_i$ (why?).

The null hypothesis is that the treatments have no effect on the outcome. If the null hypothesis holds, then the expected value of each group of observations would be $\mu$, so that $\forall i, \alpha_i = 0$. Moreover, the population variance would be $\sigma^2$.

Let the mean of the *i*th group of observations be denoted $\overline{Y_{i.}}$ and the mean of all the observations be denoted $\overline{Y_{..}}$. We denote the sum of squared deviations from the mean *within* each sample by

$$SSW = \sum_{i=1}^{I}\sum_{j=1}^{J}(Y_{ij} - \overline{Y_{i.}})^2 \qquad \textbf{(EQ 56)}$$

*SSW/I(J-1)* is an unbiased estimator of the population variance $\sigma^2$ because it sums *I* unbiased estimators $\frac{1}{J-1}\sum_{j=1}^{J}(Y_{ij} - \overline{Y_{i.}})^2$.

Similarly, we denote the sum of squared deviations from the mean *between* samples by

$$SSB = J\sum_{i=1}^{I}(\overline{Y_{i.}} - \overline{Y_{..}})^2 \qquad \textbf{(EQ 57)}$$

*SSB/(I-1)* is also an unbiased estimator of the population variance $\sigma^2$ because $\frac{1}{I-1}\sum_{i=1}^{I}(\overline{Y_{i.}} - \overline{Y_{..}})^2$ is an unbiased estimator of $\frac{\sigma^2}{J}$ (why?). So, the ratio $\frac{SSB/(\text{I-1})}{SSW/\text{I(J-1)}}$ should be 1 if the null hypothesis holds.

It can be shown that *SSB/(I-1)* is a $\chi^2$ variable with *I*-1 degrees of freedom and that *SSW/I(J-1)* is a $\chi^2$ variable with *I*(*J*-1) degrees of freedom. The ratio of two $\chi^2$ variables with *m* and *n* degrees of freedom follows a *F* distribution with (*m*,*n*) degrees of freedom. Therefore, the variable $\frac{SSB/(\text{I-1})}{SSW/\text{I(J-1)}}$ follows the *F* distribution with *(I-1, I(J-1))* degrees of freedom, and has an expected value of 1 if the null hypothesis is true.

To test the null hypothesis, we compute the value of $\frac{SSB/(\text{I-1})}{SSW/\text{I(J-1)}}$ and compare it with the critical value of an *F* variable with *(I-1, I(J-1))* degrees of freedom. If the computed value exceeds the critical value, then the null hypothesis is rejected. Intuitively, this would happen if *SSB* is 'too large', that is, there is significant variation in the sums of squares between treatments, which is what we expect when the treatment does have an effect on the observed outcome.

**Example 21: (Single factor ANOVA)**

Continuing with Example 12, assume that we have additional data for larger buffer sizes, as shown below. Can we still claim that the buffer size plays a role in determining the loss rate?

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Loss rate with 5 buffers** | 1.20% | 1.30% | 0.90% | 1.40% | 1.00% | 1.80% | 1.10% | 1.20% | 1.50% | 1.20% |
| **Loss rate with 100 buffers** | 0.10% | 0.60% | 1.10% | 0.80% | 1.20% | 0.30% | 0.70% | 1.90% | 0.20% | 1.20% |
| **Loss rate with 200 buffers** | 0.50% | 0.45% | 0.35% | 0.60% | 0.75% | 0.25% | 0.55% | 0.15% | 0.35% | 0.40% |
| **Loss rate with 500 buffers** | 0.10% | 0.05% | 0.03% | 0.08% | 0.07% | 0.02% | 0.10% | 0.05% | 0.13% | 0.04% |
| **Loss rate with 1000 buffers** | 0.01% | 0.02% | 0.01% | 0.00% | 0.01% | 0.01% | 0.00% | 0.02% | 0.01% | 0.00% |

Here, $I = 5$ and $J = 10$. We compute $\overline{Y_{5.}}$ = 1.26%, $\overline{Y_{100}}$= 0.81%, $\overline{Y_{200.}}$ = 0.44%, $\overline{Y_{500.}}$ = 0.07%, and $\overline{Y_{1000.}}$ = 0.01%. This allows us to compute $SSW = 5.13*10^{-5}$ and $SSB = 1.11*10^{-3}$. The $F$ statistic is therefore $(1.11*10^{-3}/ 4)/(5.13*10^{-5}/45)$ = 242.36. Looking up the $F$ table we find that even with only (4, 40) degrees of freedom, the critical $F$ value at the 1% confidence level is 3.83. The computed statistic far exceeds this value. Therefore, the null hypothesis is rejected.

[]

The $F$ test is somewhat anticlimactic: it only indicates that a treatment has an effect on the outcome, but it does not quantify the degree of effect. Nor does it identify whether any one treatment is responsible for the failure of the test. These questions can be resolved by *post-hoc* analysis. For example, to quantify the degree of effect, we can compute the regression of the observed effect as a function of the treatment. To identify the treatment that is responsible for the failure of the test, we can re-run the $F$ test eliminating one treatment at a time. If the $F$ test does not reject the null hypothesis with a particular treatment removed, then we can hypothesize that this treatment has a significant effect on the outcome, testing this hypothesis with a two-variable test.

If these approaches do not work, two more advanced techniques to perform multi-way comparisons are *Tukey's method* and the *Bonferroni method*. Details are beyond the scope of this text.

### 2.7.2 Multi-way layouts

It is relatively straightforward to extend the one-way layout to two or more treatments that are simultaneously applied. For instance, we may want to study the joint effect of buffer size and cross traffic workload on the loss rate. The details of this so-called *two-way layout* are beyond the scope of this text. We will merely point out that in the context of such designs, we not only have to determine the effect of a treatment on the outcome, but also deal with the possibility that only certain combinations of treatment levels affect the outcome (for instance, the combination of small buffer size and heavy cross traffic). Such *interaction effects* greatly complicate the analysis of multi-way layouts.

## *2.8 Design of experiments*

The statistically rigorous design of experiments is a complex topic. Our goal here will be to give an intuitive understanding of its essentials. Details can be found in more advanced texts devoted to the topic.

The goal of an experiment is, in the words of Sir R.A. Fisher, "...to give the facts a chance of disproving the null hypothesis." The first step in designing an experiment is to formulate a precise hypothesis that can be rejected (or not) on the basis of its results. Many experimental studies in the field of computer systems fail to meet even this obvious requirement! The careful choice of a null hypothesis cannot be over-emphasized.

Note that our analysis of hypothesis testing assumes that the elements of each sample are independently and randomly selected from the population, so that we can treat the sum of the elements of each sample as the sum of $n$ independent and identically distributed random variables. Therefore, in conducting an experiment, it is necessary to ensure that each observation is as nearly independent of the others as possible. Moreover, observations should be made so that each member of the population has an equal chance of being represented. If observation come from sampling a population, then care should be taken that no obvious bias be introduced in the sampling process.

A second consideration in the design of experiments is that enough data be collected so that the hypothesis can be conclusively rejected if necessary. To take a trivial example, it is impossible to reject the hypothesis that a coin is biased from a single coin flip. We can increase the *sensitivity* of an experiment either by collecting more observations within each sample (*enlargement*), or by collecting more samples (*repetition*).

Third, it is necessary to ensure that the experimental conditions be kept as constant be possible ('controlled') so that the underlying population does not change when making observations. Otherwise, it is impossible to determine the population whose parameters are being estimated by the statistics of the sample. For example, in a wireless network that is subject to random external interference, packet loss rates are determined not only by the signal strength of the transmitter, but also the signal strength of the interferer. If an external interferer is not controlled, a study that tries to relate a MAC data rate selection algorithm to the packet loss rate, for example, may draw incorrect conclusions.

Finally, when studying the effect of more than one treatment on the outcome of an experiment, we need to take into account the fact that the treatments may not be independent of each other. If treatments were orthogonal, we would simply need to change one treatment at a time, which reduces the analysis of the experiment to analysing a set of one-way layouts. If they are not, then we need to design a set of experiments that explores all combinations of treatments. For example, if both buffer size and cross traffic workload intensity can affect the packet loss rate at a router, and these treatments were non-orthogonal, we need to take into account the so-called interaction effects (see 2.7.2 on page 60). A trivial solution to take interactions into account is to perform a *full factorial* design, where we set up the cross product of every possible level of each treatment. For example, if we could choose between five different buffer sizes and three workload levels, we would need to conduct 15 experiments. In many cases, a full factorial design is impossible. If so, there is a considerable body of work on the design of *fractional factorial* experiments, where we may change two or more treatment levels at the same time, using statistical analysis to identify the effect of each individual treatment. These schemes can be fairly complex, in that they need to take in to account how all possible two-way, three-way,..., *n*-way interactions may affect the observed outcome. Specifically, the designs must deal with the problem of *aliasing*, that is, not being able to make out the difference between alternative combinations of treatment levels that have the same effect on the output. If certain combinations of levels can be safely ignored, or are practically unimportant, then we can greatly reduce the number of experiments without affecting the quality of the conclusions.

## *2.9 Dealing with large data sets*

In the statistical analysis of modern computer systems, all too often, the problem is not the lack of experimental data, but its surfeit. With the extensive logging of system components and the growing number of components in a system, the analyst is often confronted with the daunting task of extracting comprehensible and statistically valid results from large volumes of data. In view of this, the practice of statistical analysis-long focused on the extraction of statistically valid results from a handful of experiments-changes its character. In this section, we discuss a pragmatic approach to the analysis of large data sets based on the author’s own experiences over the last two decades. An alternative view from the perspective of computer system performance evaluation can be found in (R. Jain, The Art of Computer Systems Performance Analysis, John Wiley, 2001. Chapter 1.).

Unlike the classical notion of careful experimental design in order to test a hypothesis, the situation in contemporary systems evaluation is to focus, at least to begin with, on data *exploration*. We typically have access to a large compendium of logs and traces, and the questions we would like to answer typically fall into the following broad categories:

- How can the data help us identify the cause of poor overall performance?
- What is the relative performance of alternative implementations of one component of the system?
- Are there implicit rules that describe the data?

In answering these questions, the following procedure has proved useful.

Step 1: Extract a small sample from the entire data set and carefully read through it. Even a quick glance at the data will often point out salient characteristics that can be used to speed up subsequent analysis. Moreover, this allows a researcher to spot potential problems (certain variables not being logged, or having clearly erroneous values). Proceeding with a complex analysis in the presence of such defects only wastes time.

Step 2: Attempt to visualize the *entire* data set. For example, if every sample could be represented by a point, the entire data set could be represented by a pixellated bitmap. The human eye is quick to find non-obvious patterns, but only if presented with the entire data set. If the data set is too large, it may help to sub-sample it (taking every fifth, tenth, or hundredth sample) before visualization. Again, skipping this step will result in missing patterns that may otherwise be only revealed with considerable effort.

Step 3: Look for outliers. The presence of outliers usually indicates a deeper problem, usually either with data collection or data representation (for example, due to underflow or overflow). Usually, the removal of outliers results in the discovery of problems in the logging or tracing software, and the entire data set may have to be collected again. Even if part of the data set can be sanitized to correct for errors, it is prudent to collect the data set again.

Step 4: Formulate a preliminary null hypothesis. Choose this hypothesis with care, being conservative in your selection, so that the non-rejection of the hypothesis does not lead you to a risky conclusion.

Step 5: Use the data set to attempt to reject the hypothesis, using the techniques described earlier in this chapter.

Step 6: Frame and test more sophisticated hypotheses. Often preliminary results reveal insights into the structure of the problem whose further analysis will require the collection of additional data. The problem here is that if data is collected at different times, it is hard to control extraneous influences. The workload may have changed in the interim, or some system components may have been upgraded. Therefore, it is prudent to discard the entire prior data set to minimize the effects of uncontrolled variables. Step 6 may be repeated multiple times until the initial problem has been satisfactorily answered.

Step 7: Present and interpret the results of the analysis using appropriate graphics. An excellent source for presentation guidelines is (E. Tufte, The Visual Display of Quantitative Information, 2e, Graphics Press, 2001).

When dealing with very large data sets, where visualization is impossible, techniques derived from *data mining* and *machine learning* are often useful. We briefly outline two elementary techniques for data clustering.

The goal of a data clustering algorithm is to find hidden patterns in the data, in this case, the fact that the data can be grouped into 'clusters,' where each cluster represents 'closely' related observations. For example, in a trace of packets observed at a router interface, clusters may represent packets that fall into a certain range of lengths. A clustering algorithm automatically finds clusters in the data set that for our example, would correspond to a set of disjoint ranges of packet lengths.

A clustering algorithm takes as input a distance metric that quantifies the concept of a distance between two observations. Distances can be simple metrics, such as packet lengths, or may be more complex, such as the number of edits (that is, insertions and deletions) that need to be made to a string-valued observation to transform it into another string-valued observation. Observations within a cluster will be closer, according to the specified distance metric, than observations placed in different clusters.

In *agglomerative clustering*, we start with each observation in its own cluster. We then merge the two closest observations into a single cluster and repeat the process until the entire data set is in a single cluster. Note that to carry out repeated mergings, we need to define the distance between a point and a cluster and between two clusters. The distance between a point and a cluster can be defined as the distance from the point to the closest point in the cluster, or the average of all the all distances from that point to points in the cluster. Similarly, the distance between clusters can be defined to be the closest distance between their points or the distance between their centroids. In either case, we compute a tree such that links higher up in the tree have longer distance metrics. We can therefore truncate the tree at any point and treat the forest so created as the desired set of clusters. This approach usually does not scale beyond about 10,000 observation types on a single server; distributed computation techniques allow the processing of larger data sets.

*k-means* clustering is a technique to cluster data into *k* classes. The earliest and most widely-used algorithm for *k*-means clustering is Lloyd's algorithm. In this algorithm, we start with a set of *k* empty containers. We partition the observations into *k* sets, either randomly, or on the basis of a subsample, allocating one set to each container. We then compute the centroid of each container (this is the point that minimizes the sum of distances from all points in the set to itself). Now, each point is re-allocated to the container with the closest centroid. This may result in the container's centroid moving to a different point. We therefore re-compute the centroid for each container, re-allocating points as before. This process iterates until convergence, when no points move from one cluster to another. In most practical cases, the algorithm is found to converge after a few iterations to a globally optimal clustering. However, convergence may result in a local optimum. Several variants of this algorithm with better convergence properties are described in texts on machine learning and data mining.

## 2.10 Common mistakes in statistical analysis

We now present some common problems in statistical analysis, especially in the context of computer systems.

### 2.10.1 What is the population?

A question commonly left unanswered in statistical analyses is a precise statement of the underlying population. As we saw in 2.2 on page 30, the same sample can correspond to multiple underlying populations. It is impossible to interpret the results of a statistical analysis without carefully justifying that the sample is representative of the chosen underlying population.

### 2.10.2 Lack of confidence intervals in comparing results

Comparing the performance of two systems simply by comparing the mean values of performance metrics is an all-too-common mistake. The fact that one mean is greater than another is not statistically meaningful and may lead to erroneous conclusions. The simple solution is to always compare confidence intervals, rather than means, as described in 2.5.5 on page 45.

### 2.10.3 Not stating the null hypothesis

Although the process of research necessitates a certain degree of adjustment of hypotheses, a common problem is to carry out a statistical analysis without stating the null hypothesis. Recall that we can only reject or not reject the null hypothesis from observational data. Therefore, it is necessary to carefully formulate and clearly state the null hypothesis.

### 2.10.4 Too small a sample

If the sample size is too small, then the confidence interval associated with the sample is large, so that even a null hypothesis that is actually false will not be rejected. By computing the confidence interval around the mean during exploratory analysis, it is possible to detect this situation, and collect larger samples for populations with greater inherent variance.

### 2.10.5 Too large a sample

If the sample size is too large, then a sample that deviates even slightly from the null hypothesis will cause the null hypothesis to be rejected. This is because the confidence interval around the sample mean varies as $\frac{1}{\sqrt{n}}$. Therefore, when interpreting a test that rejects the null hypothesis, it is important to take the *effect size* into account, which is the (subjective) degree to which the rejection of the null hypothesis accurately reflects reality. For instance, suppose we hypothesize that the population mean was 0, and we found from a very large sample that the confidence interval was 0.005±0.0001. This rejects the null hypothesis. However, in the context of the problem, perhaps the value 0.005 is indistinguishable from zero, and therefore has a small ‘effect.’ In this case, we would still not reject the null hypothesis.

### 2.10.6 Not controlling all variables when collecting observations

The effect of controlling variables in running an experiment is to get a firm grasp on the nature of the underlying population. If the population being sampled changes during the course of the experiment, then the collected sample is meaningless. For example, suppose you are observing the mean delay from a campus router to a particular data centre. Suppose that during data collection, your ISP were to change their Tier 1 provider. Then, the observations made subsequent to the change would be likely to reflect a new population. It is necessary, therefore, during preliminary data analysis, to ensure that such uncontrollable effects have not corrupted the data set.

### 2.10.7 Converting ordinal to interval scales

Ordinal scales where each ordinal is numbered, such as the Likert scale (where 1 may represent ‘poor’, 2 ‘satisfactory’, 3 ‘good’, 4 ‘outstanding’ and 5 ‘excellent’), are often treated as if they are interval scales. So, if one user were to rate the streaming performance of a video player as 1 and another as 3, then the mean rating is stated to be 2. This is bad practice. It is hard to argue that the gap between ‘poor’ and ‘satisfactory’ is the same as the gap between ‘satisfactory’ and ‘good.’ Yet, that is the assumption being made when ordinal scales such as these are aggregated. In such cases, it is better to ask users to rank an experience on a linear scale from 1 to 5. This converts the ordinal scale to an interval scale and allows aggregation without making unwarranted assumptions.

### 2.10.8 Ignoring outliers

The presence of outliers should always be a cause for concern. Silently ignoring them, or deleting them from the data set altogether, not only is bad practice, but prevents the analyst from unearthing significant problems in the data collection process. Therefore, they should never be ignored.

## 2.11 Further reading

This chapter only touches on the elements of mathematical statistics. Further details can be found in texts such as (J.A. Rice). A delightfully concise summary of the basics of mathematical statistics can be found in (M.G. Bulmer, Principles of Statistics, Oliver and Boyd, 1965, re-issued by Dover, 1989). Statistical analysis is widely used in the social sciences and agriculture. The classic reference for a plethora of statistical techniques is (Snedecor and Cochran, Statistical Methods, 8e, Wiley, 1989). Exploratory data analysis is described from the perspective of a practitioner in (G. Myatt, Making Sense of Data: A Practical Guide to Exploratory Data Analysis and Data Mining, Wiley, 2006). Readers who want to learn directly from one of the masters of statistical analysis should refer to (R.A. Fisher, Statistical Methods for Research Workers, 1e, Oliver and Boyd, 1925).

## 2.12 Exercises

**1 Moments**

Prove that $\mu_3 = \mu_3' - 3\mu_2'\mu + 2\mu^3$.

**2 MGFs**

Prove that the MGF of a uniform random variable, expressed in terms of its series expansion =

$$E(e^{tx}) = \int_0^1 \left(1 + tx + \frac{(tx)^2}{2!} + \frac{(tx)^3}{3!} + \ldots\right) dx = \frac{1}{t}[e^t - 1] \quad .$$

**3 MGFs**

Prove that the $r^{th}$ moment of the uniform distribution about the origin is $1/(r+1)$.

**4 MGF of a sum of two variables**

Use MGFs to find the variance of the sum of two independent uniform standard random variables.

**5 MGF of a normal distribution**

Prove that if $X \sim N(\mu,\sigma^2)$ then $(X-\mu)/\sigma \sim N(0,1)$.

**6 Means**

Prove that the mean of a sample is the value of $x^*$ that minimizes $\sum_{i=1}^{n} (x_i - x^*)^2$

**7 Means**

Prove Equation 28.

### 8 Confidence intervals (normal distribution)

Compute the 95% confidence interval for the data values in Table 2 (reproduced below).

| Data value | Frequency |
|---|---|
| 1 | 5 |
| 2 | 7 |
| 3 | 2 |
| 7 | 2 |
| 1000 | 1 |

### 9 Confidence intervals (t distribution)

Redo Exercise 8 using the *t* distribution.

### 10 Hypothesis testing: comparing the mean to a constant

For the sample below, test the null hypothesis that the mean loss rate is 2% at the 95% confidence level.

| **Loss rate with 5 buffers** | 1.20% | 2.30% | 1.90% | 2.40% | 3.00% | 1.80% | 2.10% | 3.20% | 4.50% | 2.20% |
|---|---|---|---|---|---|---|---|---|---|---|

### 11 Chi-squared test

In Example 15, what is the value of $n_1$ beyond which the hypothesis would be rejected?

### 12 Fitting a distribution and chi-squared test

Continuing with Example 16, consider the data set below. Ignoring the first observation (i.e., (1,18)), find the best Poisson fit for the reduced sample. Use this to compute the expected count for each number of arrivals. What is the chi-squared variate value for this reduced data set? Use this to determine whether the Poisson distribution is indeed a good distribution to describe the reduced data set.

| Number of packet arrivals | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Count | 18 | 28 | 56 | 105 | 126 | 146 | 164 | 165 | 120 | 103 | 73 | 54 | 23 | 16 | 9 | 5 |

### 13 Independence, Regression, and Correlation

A researcher measures the mean uplink bandwidth of 10 desktop computers (in kbps) as well their mean number of peer-to-peer connections over the period of one hour, obtaining the following data set:

| Uplink capacity | 202 | 145 | 194 | 254 | 173 | 94 | 102 | 232 | 183 | 198 |
|---|---|---|---|---|---|---|---|---|---|---|
| # peers | 50 | 31 | 47 | 50 | 41 | 21 | 24 | 50 | 41 | 49 |

a) If the number of peers were independent of the uplink capacity, what is the expected value of the number of peers for a specific uplink capacity?

b) Can we conclude, using the chi-squared test, that the number of peers is independent of the uplink capacity, at the 95% and 99.9% confidence levels?

c) Compute the regression of the number of peers on the uplink capacity. What is the slope of the best-fit line?

d) What is the correlation coefficient between the two variates? Does this reinforce your conclusions regarding independence or dependence?

e) What portion of the variation in the value of the number of peers can be attributed to the uplink capacity?

### 14 Correlation coefficient

Prove Equation 54.

### 15 Single Factor ANOVA

A university is connected to the Internet using three ISPs. To test their relative performance, the IT staff conduct an experiment where they measured the ping times to a well-known website over each of the three providers over a period of ten days. The mean ping time using each ISP on each day is shown below. Use single-factor ANOVA to test the hypothesis that the ISPs are statistically identical.

| Day | ISP1 | ISP2 | ISP3 |
|---|---|---|---|
| 1 | 41.2 | 50.7 | 41.1 |
| 2 | 34.9 | 38.5 | 48.2 |
| 3 | 43.5 | 56.3 | 73.2 |
| 4 | 64.2 | 54.2 | 48.4 |
| 5 | 64.0 | 46.4 | 61.4 |
| 6 | 54.9 | 58.4 | 43.2 |
| 7 | 59.3 | 61.8 | 63.9 |
| 8 | 73.1 | 69.4 | 54.3 |
| 9 | 56.4 | 66.3 | 67.4 |
| 10 | 63.8 | 57.4 | 58.4 |

# CHAPTER 3 *Essentials of Linear Algebra*

## *3.1 Vectors and matrices*

Consider two runs of an experiment where a researcher collects packet traces on an Internet link. Suppose that the first trace contains 312 TCP and 39 UDP packets and that the second trace contains 432 TCP and 21 UDP packets. We can represent these results in the form of these two ordered tuples: [312, 39] and [432, 21]. Here, the positions in the tuple are implicitly associated with the meaning "TCP count" and "UDP count" respectively. We call such a representation a *vector*.

A vector is defined as an ordered set of *elements*. A vector with $n$ elements is said to have $n$ dimensions. There is a one-to-one mapping from an $n$-dimensional vector with real-valued elements to a point in an $n$-dimensional space. Returning to our example, the vector [432, 21] corresponds to a point in a space whose X and Y axes are "TCP count" and "UDP count" respectively and that has a coordinates of (432, 21). If one were to add another measurement to the tuple, say "ICMP count," then we could represent the counts in a packet trace by a vector such as [432, 21, 12] which corresponds to a point in a three- dimensional space.

Vectors can be represented in one of two ways: as *row-vectors* of the form [312, 12, 88], and as *column-vectors* of the form: $\begin{bmatrix} 312 \\ 12 \\ 88 \end{bmatrix}$. We define the zero-vector of $n$ dimensions, denoted **0,** as the vector [ 0 0 0 ... 0].

Returning to our example, we can represent packet counts in both traces simultaneously using an array that looks like this:

$$\begin{bmatrix} 312 & 39 \\ 432 & 21 \end{bmatrix}$$

Such a representation is called a *matrix*.

Unlike a vector, whose elements may be unrelated, elements in the same column of a matrix are usually related to each other; in our example, all elements in the first column are TCP counts. In general, an array with *m* rows and *n* columns is called an $m \times n$ matrix.The element in the *i*th row and *j*th column of a matrix named *A* is usually represented by the symbol $a_{ij}$. In the example above, $a_{12}$ is 39 and $a_{21}$ is 432.

Although vector and matrix representations can be used for arbitrary element types, such as for character strings, in our discussion, we will assume that the elements of a vector or a matrix are members of a mathematical *field*. A field *F* is a finite or infinite *set* along with the *operations* of addition (denoted '+') and multiplication (denoted '*') on elements of this set that satisfy the following axioms:

1. Closure under addition and multiplication: For *a*, *b* in *F*, if $a+b = c$ and $a*b=d$, then *c* and *d* are also in *F*.
2. Commutativity of addition and multiplication: For *a*, *b* in *F*, if $a+b = b+a$ and $a*b=b*a$.
3. Associativity of addition and multiplication: For *a*, *b, c* in *F*, $(a+b) +c = a + (b+c)$.
4. Existence of distinct additive and multiplicative identity elements in the set: There are distinct elements denoted '0' and '1' in *F*, such that for all *a* in *F*, $a+0 = a$ and $a*1 = a$.
5. Existence of additive and multiplicative inverses: For every *a* in *F* there is an element *b* also in *F* such that $a+b = 0$. For every *a* in *F* other than '0', there is an element *also* in *F* such that $a*c = 1$.
6. Distributivity of multiplication over addition: For all *a, b* and *c* in *F*, the following equality holds: $a *(b+c) = (a*b) + (a*c)$.

# 3.2 Vector and matrix algebra

In this section, we will study some basic operations on vectors and matrices.

## 3.2.1 Addition

The sum of two vectors of the same dimension is a vector whose elements are the sums of the corresponding elements of each vector. Addition is not defined for vectors with different dimensions.

The sum of two matrices with the same number of rows and columns is a matrix whose elements are the sums of the corresponding elements of each matrix. Addition is not defined for matrices with different numbers of rows or columns.

Because the elements of a vector or a matrix are drawn from a field and vector and matrix addition operates element by element, vector and matrix addition obeys the standard field properties of closure, commutativity, associativity, the existence of inverse and identity elements, and distributivity. In other words, vectors and matrices whose elements are drawn from a particular field themselves form fields.

## 3.2.2 Transpose

The transpose of a row-vector *x*—denoted $x^T$—is a column-vector whose *j*th row is the *i*th column of *x*. The transpose of an $m \times n$ matrix *A*, denoted $A^T$ is an $n \times m$ matrix whose [*j, i*]th element—denoted $a_{ji}$—is the [*i, j*]th of *A*, $a_{ij}$.

**Example 1: (Transpose)**

The transpose of [1,3,5,1] is $\begin{bmatrix} 1 \\ 3 \\ 5 \\ 1 \end{bmatrix}$. The transpose of $\begin{bmatrix} 23 & 2 & 9 \\ 98 & 7 & 89 \\ 34 & 9 & 1 \end{bmatrix}$ is $\begin{bmatrix} 23 & 98 & 34 \\ 2 & 7 & 9 \\ 9 & 89 & 1 \end{bmatrix}$.

[]

### 3.2.3 Multiplication

Multiplying a vector *x* by a real number (or *scalar*) *s* results in the multiplication of each element (i.e., *scaling*) of the vector by that real. That is,

$$s[x_1, x_2, \ldots, x_n] = [sx_1, sx_2, \ldots, sx_n] \tag{EQ 1}$$

Similarly, multiplying a matrix *A* by a real number (scalar) *s* results in the multiplication of each element of the matrix by that real. That is,

$$s\begin{bmatrix} a_{11} & \ldots & a_{1n} \\ \ldots & \ldots & \ldots \\ a_{m1} & \ldots & a_{mn} \end{bmatrix} = \begin{bmatrix} sa_{11} & \ldots & sa_{1n} \\ \ldots & \ldots & \ldots \\ sa_{m1} & \ldots & sa_{mn} \end{bmatrix} \tag{EQ 2}$$

The product of two vectors can be defined either in terms of a *dot* product or a *cross* product. The dot product of vector *x* with elements $x_i$ and vector *y* with elements $y_i$ is defined as the scalar *s* obtained as the sum of the element-by-element product. That is,

$$s = x.\mathrm{y} = \sum_{i=1}^{n} x_i y_i \tag{EQ 3}$$

The dot product is undefined if the two vectors do not have the same dimension.

The *cross* product of two vectors is not relevant to computer networking and will not be discussed further.

Unlike the dot product of two vectors, which is a scalar, the product of two matrices is a matrix whose [*i*, *j*]th element is the dot product of the *i*th row of the first matrix and the *j*th column of the second matrix. That is, if *C = AB*, then

$$c_{ij} = \sum_{k=1}^{n} a_{ik} y_{kj} \tag{EQ 4}$$

Note that the number of columns in *A* (the dimension of each row of *A*) must equal the number of rows in *B* (the dimension of each column in *B*). Thus, the product of an $m \times n$ matrix by an $n \times o$ matrix results in an $m \times o$ matrix. Therefore, the product of a *n* dimensional row-vector-a matrix of size $1 \times n$ -with an $n \times n$ matrix is a row-vector of dimension *n*.

**Example 2: (Matrix multiplication)**

The product of $\begin{bmatrix} 23 & 2 & 9 \\ 98 & 7 & 89 \\ 34 & 9 & 1 \end{bmatrix}$ and $\begin{bmatrix} 2 & 5 & -2 \\ 4 & 9 & 8 \\ 3 & 0 & 1 \end{bmatrix}$ is $\begin{bmatrix} 81 & 133 & -21 \\ 491 & 553 & -51 \\ 107 & 251 & 5 \end{bmatrix}$. To obtain $c_{11}$, for example, we compute 23.2 + 2.4 + 9.3 = 46 + 8 + 27 = 81.

[]

Matrix multiplication is associative, that is *(AB)C = A(BC)*, but it is not commutative. That is,

$$AB \neq BA \tag{EQ 5}$$

This follows trivially from the definition of multiplication, or from the observation that although *AB* may be defined, if the number of columns in *B* differs from the number of rows in *A,* then *BA* may not even be defined. Moreover, if *AB=0*, it is not necessary that either *A* or *B* be the null matrix. This is unlike the case with scalars, where *ab=0* implies that one of *a* or *b* is zero. As a corollary, if *AB=AC,* then *B* does not necessarily have to be the same as *C*.

### 3.2.4 Square matrices

A matrix $A$ is *square* if it has the same number of rows and columns.

An $n \times n$ square matrix $I$ with '1' along the main diagonal and '0' elsewhere has the property that multiplication of any $n \times n$ square matrix $A$ with this matrix does not change $A$. That is, $AI = IA = A$. Hence, $I$ is called the *identity* matrix.

### 3.2.5 Exponentiation

If a matrix $A$ is square then its product with itself is defined and denoted $A^2$. If $A$ has $n$ rows and columns, so does $A^2$. By induction, all higher powers of $A$ are also defined and also have $n$ rows and columns.

**Example 3: (Exponentiation)**

Let $A$ be $\begin{bmatrix} 5 & 0 & 0 \\ 0 & 7 & 0 \\ 0 & 0 & 2 \end{bmatrix}$ . Then, $A^2 = \begin{bmatrix} 25 & 0 & 0 \\ 0 & 49 & 0 \\ 0 & 0 & 4 \end{bmatrix}$ and $A^3 = \begin{bmatrix} 125 & 0 & 0 \\ 0 & 343 & 0 \\ 0 & 0 & 8 \end{bmatrix}$ . Finding the higher powers of $A$ in this example is particularly straightforward (Why?). In general, if $A$ is a *diagonal* matrix, that is, it has zeroes in every position except the diagonal, then it is easy to show that the $(i,i)^{\text{th}}$ element of the $k$th power of $A$ is $a_{ii}{}^k$.

[]

**Example 4: (Exponentiation of an identity matrix)**

Prove that if $A^2 = A$ then $A = I$. That is, the only matrix whose higher powers are the same as the matrix itself is the identity matrix.

Because $A^2 = A$, we can pre-multiply both sides with the matrix $A^{-1}$ to give the identity $A^{-1}A^2=A^{-1}A$. Expanding the left hand side, we get $A^{-1}AA = IA = A$. The right hand side evaluates to $I$, which establishes the identity. QED.

[]

## 3.3 Linear combinations, independence, basis, and dimension

This section introduces some important concepts that will be used in later sections.

### 3.3.1 Linear combinations

Consider a set of $k$ real-valued variables $x_1, x_2,...,x_k$. Suppose we are given a set of $k$ real-valued weights $w_1, w_2,...,w_k$. Then, we can define the weighted sum of these variables as $s = w_1x_1 + w_2x_2 +...+ w_kx_k$. This sum 'mixes' the variables in proportion to the weights. Moreover, the mixing is linear, in that each variable's contribution to the sum is expressed as a product with another real (rather than, say, as $e^wx$). Therefore, we call $s$ a *linear combination* of the variables.

We can generalize the notion of linear combination to vectors. Here, each $x_i$ as a vector, so that their linear combination, $s$, is also a vector. Of course, each vector must have the same number of elements. Note that each component of $s$ is a linear combination of the corresponding elements of the underlying vectors.

**Example 5: (Linear combination of scalars)**

Compute the linear combination of the scalars 2, 4, 1, 5 with weights 0.1, 0.4, 0.25, 0.25.

The linear combination is 0.1*2 + 0.4*4 + 0.25*1 + 0.25*5 = 0.2 + 1.6 + 0.25 + 1.25 = 3.3.

[]

**Example 6: (Linear combination of vectors)**

Compute the linear combination of the vectors [2 4 1 5], [3 5 1 2], [5 6 2 1], [9 0 1 3] with weights 0.1, 0.4, 0.25, 0.25.

The linear combination is given by 0.1*[2 4 1 5] + 0.4*[3 5 1 2] + 0.25*[5 6 2 1] + 0.25*[9 0 1 3]. Clearly, the first element of *s* is given by 0.1*2 + 0.4*3 + 0.25*5 + 0.25*9 = 0.2 + 1.2 + 1.25 + 2.25 = 4.9. Similarly, the other elements are 5.1, 1.25 and 2.3, so that *s* = [4.9 5.1 1.25 2.3].

[]

### 3.3.2 Linear independence

Consider a set of *k* vectors $x_1, x_2,...,x_k$. Suppose we can express one of the vectors, say $x_i$,as a linear combination of the others. Then, the value of $x_i$ *depends* on the others: if the remaining vectors assume certain values, then the value of $x_i$ is known and cannot be chosen arbitrarily. This means that we have removed some degrees of freedom in assigning arbitrary values to the vectors.

Specifically, suppose we can express $x_i$ as a linear combination of the remaining *k-1* vectors using an appropriately chosen set of *k-1* weights. Then, we can write

$$x_i = w_1x_1 +... + w_{i-1}x_{i-1} + w_{i+1}x_{i+1} +... +w_kx_k \qquad \text{(EQ 6)}$$

Or, transposing terms:

$$w_1x_1 +... + w_{i-1}x_{i-1} - x_i + w_{i+1}x_{i+1} +... +w_kx_k = \mathbf{0} \qquad \text{(EQ 7)}$$

This motivates the following definition of independence of a set of vectors: we say that a set of vectors is independent if the only set of weights that satisfies Equation 7 is $w$=**0.**

Note that if a set of vectors is not linearly independent, *any* one of them can be rewritten in terms of the others (Why?).

**Example 7: (Linear independence)**

The three vectors:

$$x_1 = [3\ 0\ 2]$$

$$x_2 = [-3\ 21\ 12]$$

$$x_3 = [21\ -21\ 0]$$

are not linearly independent because $6x_1 - x_2 - x_3 = \mathbf{0}$. The first and third vectors are independent because the third element of the first vector cannot be generated by the third vector (Why?).

[]

Later, we will demonstrate that if a set of vectors is linearly independent, then the matrix formed by the vectors is *non-singular*, that is, has a non-zero *determinant*.

### 3.3.3 Vector spaces, basis, and dimension

Given a set of $k$ vectors, suppose we can identify a subset of $r$ vectors that linearly independent. That is, the remaining vectors can be written as linear combinations of the $r$ vectors. Then, we call these $r$ vectors the *basis* set of this set of vectors. They form the essential core from which we can derive the rest of the vectors. In this sense, the remaining vectors can be thought to be redundant. For instance, in Example 7, the first and third vectors constitute a basis, and the second vector can be generated from the basis set as $x_2 = 6x_1 - x_3$.

We can now generalize this observation as follows. Suppose we are given a set of $r$ linearly independent vectors. What is the set of vectors that can be generated as linear combinations of this set? Clearly, there are an infinite number of such vectors. We call this infinite set a *vector space* generated by the basis set (note that a 'vector space' is a precisely defined mathematical object - this is only an informal definition of the concept). The number of basis vectors (the cardinality of the basis set) is called the *dimension* of this space.

**Example 8: (Basis and dimension)**

A simple way to guarantee that a set of vectors is linearly independent is to set all but one element in each vector to zero. For instance, the vectors $x_1$ = [1 0 0], $x_2$ = [0 1 0], $x_3$ = [0 0 1] are guaranteed to be linearly independent (Why?). Let us find the vector space generated by this basis set. Consider an arbitrary vector $x = [a\ b\ c]$. This vector can be expressed as linear combination of the basis set as $x = ax_1 + bx_2 + cx_3$. Therefore, this basis set generates the vector space of all possible vectors with three real-valued elements.

If we think of a vector with three real-valued elements as corresponding to a point in three-dimensional space, where the elements of the vector are its Cartesian coordinates, then the basis vectors generate all possible points in three-dimensional space. It is easy to see that the basis vectors correspond to the three ordinal axes. It should now be clear why we call the generated vectors a 'space', and why the cardinality of the basis set is the dimensionality of this space.

[]

## *3.4 Solving linear equations using matrix algebra*

We now turn our attention to a very important application of matrix algebra, which is to solve sets of linear equations.

### 3.4.1 Representation

Systems of linear equations are conveniently represented by matrices. Consider the set of linear equations:

$$\begin{aligned} 3x + 2y + z &= 5 \\ -8x + y + 4z &= -2 \\ 9z + 0.5y + 4z &= 0.9 \end{aligned} \qquad \text{(EQ 8)}$$

We can represent this set of equations by the matrix

$$\begin{bmatrix} 3 & 2 & 1 & 5 \\ -8 & 1 & 4 & -2 \\ 9 & 0.5 & 4 & 0.9 \end{bmatrix} \qquad \text{(EQ 9)}$$

where the position of a number in the matrix implicitly identifies it either as a coefficient of a variable or a value on the right hand side. This representation can be used for any set of linear equations. If the rightmost column is **0,** then the system is said to be *homogeneous*.The submatrix corresponding to the left hand size of the linear equations is called the *coefficient matrix.*

### 3.4.2 Elementary row operations and Gaussian elimination

Given a set of equations, certain simple operations allow us to generate new equations. For example, multiplying the left- and right-hand sides of any equation by a scalar generates a new equation. Moreover, we can add or subtract the left- and right-hand sides of any pair of equations to also generate new equations.

In our example above, the first two equations are $3x + 2y + z = 5$ and $-8x + y + 4z = -2$. We can multiply the first equation by 3 to get the new equation $9x + 6y + 3z = 15$. We can also add the two equations to get a new equation $(3-8)x + (2+1)y + (1+4)z = (5-2)$, which gives us the equation $-5x + 3y + 5z = 3$.

We can also combine these operations. For example, we could multiply the second equation by 2 and subtract it from the first one like this:

$$\begin{aligned}(3-(-16))x + (2-2)y + (1-8)z &= 5-(-4)\\ 19x - 7z &= 9\end{aligned}$$

This results in an equation where the variable $y$ has been *eliminated* (i.e., does not appear). We can similarly multiply the third equation by 4 and subtract it from the first one to obtain another equation that also eliminates $y$. We now have two equations in two variables that we can trivially solve to obtain $x$ and $z$. Putting their values back into any of the three equations allows us to find $y$.

This approach, in essence, is the well-known technique called *Gaussian elimination*. In this technique, we pick any one variable and use multiplications and additions on the set of equations to eliminate that variable from all but one equation. This transforms a system with *n* variables and *m* equations to a system with *n-1* variables and *m-1* equations. We can now recurse to obtain, in the end[1], an equation with one variable, which solves the system for that variable. By substituting this value back into the reduced set of equations, we solve the system.

When using a matrix representation of the set of equations, the elementary operations of multiplying an equation by a scalar and of adding two equations correspond to two *row operations*. The first row operation multiplies all the elements of a row by a scalar and the second row operation is the element-by-element addition of two rows. It is easy to see that these are exactly analogous to the operations in the previous paragraphs. The Gaussian technique uses these elementary row operations to manipulate the matrix representation of a set of linear equations so that one row looks like this: [0 0 ... 0 1 0 ... 0 *a*]. This allows us to read off the value of that variable. We can use this to substitute for this variable in the other equations, so that we are left with a system of equations with one less unknown, and, by recursion, to find the values of all the variables.

**Example 9: (Gaussian elimination)**

Use row operations and Gaussian elimination to solve the system given by $\begin{bmatrix} 3 & 2 & 1 & 5 \\ -8 & 1 & 4 & -2 \\ 9 & 0.5 & 4 & 0.9 \end{bmatrix}$.

We subtract row 3 from row 2 to obtain $\begin{bmatrix} 3 & 2 & 1 & 5 \\ -17 & 0.5 & 0 & -2.9 \\ 9 & 0.5 & 4 & 0.9 \end{bmatrix}$. We then subtract 0.25 times row 4 from row 1 to obtain

$\begin{bmatrix} 0.75 & 1.875 & 0 & 4.775 \\ -17 & 0.5 & 0 & -2.9 \\ 9 & 0.5 & 4 & 0.9 \end{bmatrix}$. Note that the first two rows represent a pair of equations in two unknowns. We multiply the second row

by 1.875/0.5 = 3.75 and subtract from the first row to obtain $\begin{bmatrix} 64.5 & 0 & 0 & 15.65 \\ -17 & 0.5 & 0 & -2.9 \\ 9 & 0.5 & 4 & 0.9 \end{bmatrix}$. This allows us to read off $x$ as 15.65/66.525 = 0.2426. Substituting this into row 2, we get -17*0.2426 + 0.5$y$ = -2.9, which we solve to get $y$ = 2.4496. Substituting this

[1] Assuming that the equations are self-consistent and have at least one solution. More on this below.

into the third row, we get 9*0.2426 + 0.5* 2.4496 + 4*z* = 0.9, so that *z* = 0.6271. Checking, 3*0.2426 + 2*2.4484 - 0.6271 = 4.9975, which is within rounding error of 5.

[]

In practice, choosing which variable to eliminate first has important consequences. Choosing a variable unwisely may require us to maintain matrix elements to very high degrees of precision, which is costly. There is a considerable body of work on algorithms to carefully choosing the variables to eliminate, which are also called the *pivots*. Standard matrix packages, such as MatLab, implement these algorithms.

### 3.4.3 Rank

So far, we have assumed that a set of linear equations always has a consistent solution. This is not always the case. A set of equations has no solution or has an infinite number of solutions if it is either *over-determined* or *under-determined* respectively. A system is over-determined if the same variable assumes inconsistent values. For example, a trivial over-determined system is the set of equations: $x = 1$ and $x = 2$. Gaussian elimination will fail for such systems.

A system is under-determined if it admits more than one answer. A trivial instance of an under-determined system is the system of linear equations: $x + y = 1$. Here, we can choose an infinite number of values of *x* and *y* that satisfy this equation. Gaussian elimination on such a system results in some set of variables expressed as linear combinations of the independent variables. Each assignment of values to the independent variables will result in finding a consistent solution to the system.

Given a system of *m* linear equations using *n* variables, the system is under-determined if $m < n$. If *m* is at least as large as *n*, the system may or may not be under-determined, depending on whether some equations are ‘repeated.’ Specifically, we define an equation as being *linearly dependent* on a set of other equations if it can be expressed as a linear combination of the other equations (the vector corresponding to this equation is a linear combination of the vectors corresponding to the other equations). If one equation in a system of linear equations is linearly dependent on the others, then we can reduce the equation to the equation *0=0* by a suitable combination of multiplications and additions. Thus, this equation does not give us any additional information and can be removed from the system without changing the solution.

If of *m* equations in a system, *k* can be expressed as a linear combination of the other *m-k* equations, then we really only have *m-k* equations to work with. This value is called the *rank* of the system, denoted *r*. If $r < n$, then the system is under-determined. If $r=n$, then there is only one solution to the system. If $r > n$, then the system is over-determined, and therefore inconsistent. Note that the rank of a matrix is the same as the cardinality of the basis set of the corresponding set of row vectors.

**Example 10: (Rank)**

We have already seen that the system of equations $\begin{bmatrix} 3 & 2 & 1 & 5 \\ -8 & 1 & 4 & -2 \\ 9 & 0.5 & 4 & 0.9 \end{bmatrix}$ has a unique assignment of consistent values to the variables *x*, *y*, and *z*. Therefore, it has a rank of 3.

Consider the system $\begin{bmatrix} 3 & 2 & 1 & 5 \\ -8 & 1 & 4 & -2 \\ 6 & 4 & 3 & 10 \end{bmatrix}$ . We see that the third row is just the first row multiplied by 2. Therefore, it adds no additional information to the system and can be removed. The rank of this system is 2 (it is under-determined), and the resultant system has an infinity of solutions.

Now, consider the system $\begin{bmatrix} 3 & 2 & 1 & 5 \\ -8 & 1 & 4 & -2 \\ 9 & 0.5 & 4 & 0.9 \\ 3 & 2 & 1 & 4 \end{bmatrix}$. We know that the first three rows are linearly independent and have a rank of 3. The fourth row is chosen to inconsistent with the first row, so the system is over-determined, and has no solution. The resulting system has a rank of 4.

[]

Many techniques are known to determine the rank of a system of equations. These are, however, beyond the scope of this discussion. For our purpose, it suffices to attempt Gaussian elimination, and report a system to be over-determined, that is, have a rank of at least *n* if an inconsistent solution is found, and to be under-determined, that is, with a rank smaller than *n*, if an infinite number of solutions can be found. If the system is under-determined, the rank is precisely the number of equations that do not reduce to the trivial equation 0=0.

### 3.4.4 Determinants

The determinant of a matrix is, of itself, of not much practical value, although, as we will see, the solution of a set of linear equations can be compactly represented using them. However, it is necessary to understand the concept as a preliminary to studying eigenvalue problems.

The determinant $D$ = det $A$ of a two-by-two matrix is a *scalar* defined as follows:

$$D = detA = det\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = \begin{vmatrix} a_{11}a_{12} \\ a_{21}a_{22} \end{vmatrix} = a_{11}a_{22} - a_{12}a_{21} \qquad \textbf{(EQ 10)}$$

Note the use of vertical lines (instead of brackets) to indicate the determinant of the matrix, rather than the matrix itself. The determinant of a two-by-two matrix is called a determinant of *order* 2.

To describe the determinant of a larger (square) matrix, we will need the concept of *submatrix* corresponding to an element $a_{jk}$. This is the matrix $A$ from which the $j^{\text{th}}$ row and the $k^{\text{th}}$ column have been deleted and is denoted $S_{jk}(A)$. The determinant of this submatrix, i.e., $det\ S_{jk}(A) = |S_{jk}(A)|$ is called the *minor* of $a_{jk}$ and is denoted $M_{jk}$.

Note that the submatrix of a matrix has one fewer row and column. The determinant of a *n*-by-*n* matrix has order *n*. Therefore, each of its minors has an order *n*-1.

We now define another auxiliary term, which is the *co-factor* of $a_{jk}$ denoted $C_{jk}$.This is defined by

$$C_{jk} = (-1)^{j+k}M_{jk} \qquad \textbf{(EQ 11)}$$

We are now in a position to define the determinant of a matrix. The determinant of a matrix is defined recursively as follows:

$$D = \sum_{j=1}^{n} a_{ij}C_{ij} = \sum_{k=1}^{n} a_{ki}C_{ki} \qquad \textbf{(EQ 12)}$$

where $i$ is an arbitrary row or column. It can be shown that $D$ does not change no matter which column or row is chosen for expansion. Moreover, it can be shown (see the Exercises) that the determinant of a matrix does not change if the matrix is transposed. That is, $|A| = |A^T|$.

**Example 11: (Determinants)**

Compute the determinant of the matrix $\begin{bmatrix} 2 & 5 & -2 \\ 4 & 9 & 8 \\ 3 & 0 & 1 \end{bmatrix}$.

We will compute this by expanding the third row, so that we can ignore the middle co-factor corresponding to the element $a_{32} = 0$. The determinant is given by

$$a_{31}C_{31} + a_{33}C_{33} = 3(-1)^{3+1}M_{31} + 1(-1)^{3+3}M_{33} = 3\begin{vmatrix} 5 -2 \\ 98 \end{vmatrix} + 1\begin{vmatrix} 25 \\ 49 \end{vmatrix} = 3(40-(-18)) + 1(18-20) = 174 + (-2) = 172$$

As a check, we expand by the center column to obtain

$$a_{12}C_{12}+a_{22}C_{22} = 5(-1)^{1+2}M_{12}+1(-1)^{2+2}M_{22} = -5\begin{vmatrix}48\\31\end{vmatrix}+9\begin{vmatrix}2-2\\31\end{vmatrix} = -5(4-24)+9(2-(-6)) = 100+72 = 172$$

[]

Because we can compute a determinant using any row or column, it is easy to see that if a matrix has a zero column or row, then the corresponding determinant is 0.

It can be easily shown that a square matrix with *n* rows and columns has rank *n* if and only it has a non-zero determinant. Moreover, a square matrix has an inverse (is *non-singular)* if and only if has a non-zero determinant.

### 3.4.5 Cramer's theorem

Computing the determinant of a matrix allows us to (in theory, at least) trivially solve a system of equations. In practice, computing the determinant is more expensive than Gaussian elimination, so *Cramer's rule*, discussed below is useful mostly to give us insight into the nature of the solution.

Cramer's theorem states that if a system of *n* linear equations in *n* variables $Ax = B$ has a non-zero coefficient determinant $D = det\ A$, then the system has precisely one solution, given by

$$x_i = D_i/D$$

where $D_i$ is determinant of a matrix obtained by substituting $B$ for the *i*th column in $A$. Thus, if we know the corresponding determinants, we can directly compute the $x_i$s using this theorem (this is often called Cramer's rule). Note that if the system is homogeneous, that is, $B$=**0**, then each of the $D_i$s is zero (Why?), so that each of the $x_i$s is also 0.

If the determinant of the coefficient matrix $A$, i.e., $D$, is 0, and the system is homogeneous, then Cramer's rule assigns each variable the indeterminate quantity 0/0. However, it can be shown that if the system of equations is homogeneous and $D$=0*,* then the system does, in fact, have non-zero solutions. This important fact is the point of departure for the computation of the eigenvalues of a matrix.

### 3.4.6 The inverse of a matrix

The inverse of a square matrix $A$ denoted $A^{-1}$ is a matrix such that $AA^{-1} = A^{-1}A = I.$

**Example 12: (Inverse)**

Prove that the inverse of a matrix is unique.

If $A$ had an inverse $B$ as well as an inverse $C$, then $AB=BA=AC=CA=I.$ So, $B = BI = B(AC) = (BA)C = IC = C.$

[]

Not all square matrices are invertible: a matrix that does not have an inverse is called a *singular* matrix. All singular matrices have a determinant of zero. If a matrix is not singular, its inverse is given by:

$$A^{-1} = \frac{1}{|A|}[C_{jk}]^T = (1/|A|)\begin{bmatrix} C_{11} & C_{21} & \cdots & C_{n1} \\ C_{12} & C_{22} & \cdots & C_{n2} \\ \cdots & \cdots & \cdots & \cdots \\ C_{1n} & C_{2n} & \cdots & C_{nn} \end{bmatrix} \qquad \textbf{(EQ 13)}$$

where $C_{jk}$ is the co-factor of $a_{jk}$. Note that the co-factor matrix is transposed as compared to $A$. As a special case, the inverse of a two-by-two matrix

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \quad \text{is} \quad A^{-1} = \frac{1}{|A|}\begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix} \tag{EQ 14}$$

**Example 13: (Inverse)**

Compute the inverse of the matrix $\begin{bmatrix} 2 & 3 \\ 6 & 2 \end{bmatrix}$.

The determinant of the matrix is 2*2 - 6*3 = -14. We can use Equation 14 to compute the inverse as:

$\frac{1}{-14}\begin{bmatrix} 2 & -3 \\ -6 & 2 \end{bmatrix}$.

[]

## 3.5 Linear transformations, eigenvalues and eigenvectors

This section deals with the practically important problem of linear transformations and their computation using eigenvalues and eigenvectors.

### 3.5.1 A matrix as a linear transformation

Recall that the product of an $n \times n$ matrix with an *n* dimensional column vector--a matrix of size $n \times 1$ --is another column vector of dimension *n*. We can therefore view the matrix as *transforming* the input vector into the output vector.

Note that the $k^{th}$ element of the output column vector is formed by combining all the elements of the input vector using the weights found in the $k^{th}$ row of the matrix. Recall that this is a *linear combination* of elements of the input vector. A square matrix represents (the weights corresponding to) a set of such linear combinations and thus is said to represent a *linear* transformation of the input vector.

**Example 14: (Matrix as a linear transformation)**

Consider the matrix $\begin{bmatrix} 2 & 3 \\ 6 & 2 \end{bmatrix}$ and an input vector $\begin{bmatrix} a \\ b \end{bmatrix}$. The multiplication of this vector with the matrix, i.e., $\begin{bmatrix} 2 & 3 \\ 6 & 2 \end{bmatrix} * \begin{bmatrix} a \\ b \end{bmatrix}$ is the output vector $\begin{bmatrix} 2a + 3b \\ 6a + 2b \end{bmatrix}$. The first element of the output vector can be thought of as combining the input elements *a* and *b* with weights 2 and 3 (the first row of the matrix), and the second element of the output vector can be thought of as combining the inputs with weights 6 and 2 (the second row of the matrix).

[]

The definition of matrix multiplication allows us to represent the composition of two linear transformations as a matrix product. Specifically, suppose that the matrix *A* transforms a column vector *x* to another column vector *x'* and that the matrix *B* is now used to transform *x'* to another vector *x''*. Then, *x''=Bx' = B(Ax) = (BA)x = Cx*, where *C=BA* is the product of the two transformation matrices. That is, we can represent the composition of the two transformations as the matrix product of the transformation matrices.

**Example 15: (Composition of linear transformations)**

We have already seen that the matrix $\begin{bmatrix} 2 & 3 \\ 6 & 2 \end{bmatrix}$ transforms a vector $\begin{bmatrix} a \\ b \end{bmatrix}$ to the vector $\begin{bmatrix} 2a+3b \\ 6a+2b \end{bmatrix}$. Suppose that we apply the matrix $\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$ to this output. The resultant value is $\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} * \begin{bmatrix} 2a+3b \\ 6a+2b \end{bmatrix} = \begin{bmatrix} 4a+6b \\ 12a+4b \end{bmatrix}$. Instead, we can compute the product of the two transformation matrices as $\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} * \begin{bmatrix} 2 & 3 \\ 6 & 2 \end{bmatrix} = \begin{bmatrix} 4 & 6 \\ 12 & 4 \end{bmatrix}$. Then, applying this product to the initial input gives us $\begin{bmatrix} 4 & 6 \\ 12 & 4 \end{bmatrix} * \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} 4a+6b \\ 12a+4b \end{bmatrix}$.

[]

### 3.5.2 The eigenvalue of a matrix

Consider the matrix $\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$ and an input vector $\begin{bmatrix} a \\ b \end{bmatrix}$. The multiplication of this vector with the matrix is the output vector $\begin{bmatrix} 2a \\ 2b \end{bmatrix} = 2* \begin{bmatrix} a \\ b \end{bmatrix}$. Therefore, the matrix represents a *doubling* transformation on its input: the result of applying this matrix to a vector is equivalent to multiplying the vector by the scalar 2.

When the result of a matrix multiplication with a *particular* vector is the same as a scalar multiplication with that vector, we call the scalar an *eigenvalue* of the matrix and the corresponding vector an *eigenvector*. More precisely, we define an eigenvalue of a square matrix *A* to be a scalar $\lambda$ such that, for *some* non-zero column vector *x*,

$$Ax = \lambda x \tag{EQ 15}$$

The magnitude of an eigenvalue indicates the degree to which the matrix operation scales an eigenvector: the larger the magnitude, the greater the scaling effect.

**Example 16: (Eigenvalues and eigenvectors)**

Compute the eigevalues and corresponding eigenvectors of the matrix $\begin{bmatrix} 4 & 6 \\ 12 & 4 \end{bmatrix}$.

Let the eigenvector be $\begin{vmatrix} x_1 \\ x_2 \end{vmatrix}$. Then, $\begin{bmatrix} 4 & 6 \\ 12 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \lambda \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$. Expanding the left hand side, we get a system of two equations:

$$\begin{aligned} 4x_1 + 6x_2 &= \lambda x_1 \\ 12x_1 + 4x_2 &= \lambda x_2 \end{aligned}$$

Transposing terms, we get

$$\begin{aligned} (4-\lambda)x_1 + 6x_2 &= 0 \\ 12x_1 + (4-\lambda)x_2 &= 0 \end{aligned} \tag{EQ 16}$$

We recognize this as a homogenous system of two linear equations. Recall from Section 3.4.5 on page 76 that this system has a non-zero solution only if the determinant of the coefficient matrix is zero. That is,

$$\begin{vmatrix} (4-\lambda) & 6 \\ 12 & (4-\lambda) \end{vmatrix} = 0$$

We can expand this to get the quadratic:

$$(4-\lambda)^2 - 72 = 0$$

which we solve to find:

$$(4-\lambda)^2 = 72$$

$$\lambda = 4 \pm \sqrt{72}$$

Given these two eigenvalues, we compute the corresponding eigenvectors as follows. To begin with, we substitute the value $\lambda = 4+\sqrt{72}$ in Equation 16 to get:

$$(4-4-\sqrt{72})x_1 + 6x_2 = 0$$
$$12x_1 + (4-4-\sqrt{72})x_2 = 0$$

$$-\sqrt{72}x_1 + 6x_2 = 0$$
$$12x_1 - \sqrt{72}x_2 = 0$$

Since the coefficient matrix has a zero determinant (by design!) we know that it has a rank at most 1. It has a rank at least 1, by definition, so the rank is identically 1. This means that we have an infinite number of solution eigenvectors, parametrized by a single free variable. This is represented as $\begin{bmatrix} x_1 \\ \sqrt{72}/6x_1 \end{bmatrix} = \begin{bmatrix} x_1 \\ \sqrt{2}x_1 \end{bmatrix}$. For instance, if we set $x_1 = 1$, then one possible eigenvector is $\begin{bmatrix} 1 \\ \sqrt{2} \end{bmatrix}$. This can also be represented as $a \begin{bmatrix} 1 \\ \sqrt{2} \end{bmatrix}$, where $a$ is an arbitrary scalar.

As a check, note that $\begin{bmatrix} 4 & 6 \\ 12 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ \sqrt{2} \end{bmatrix} = \begin{bmatrix} 4+6\sqrt{2} \\ 12+4\sqrt{2} \end{bmatrix}$ and $(4+\sqrt{72}) \quad * \begin{bmatrix} 1 \\ \sqrt{2} \end{bmatrix} = \begin{bmatrix} 4+6\sqrt{2} \\ 12+4\sqrt{2} \end{bmatrix}$.

We interpret this geometrically as follows. Suppose we represent the Cartesian coordinates of a point $(x, y)$ by the vector $\begin{bmatrix} x \\ y \end{bmatrix}$. Then, the eigenvectors parametrized by $x_1$ as $\begin{bmatrix} x_1 \\ \sqrt{2}x_1 \end{bmatrix}$ are the set of points that lie on the line $y = \sqrt{2}x$. Points that lie on this line are transformed by the matrix to other points *also on the same line*, because the effect of the matrix on its eigenvector is to act like a scalar, which does not change the direction of a vector. Moreover, the *degree* of scaling is the associated eigenvalue of $4+\sqrt{72}$. That is, a point a unit distance from the origin and on this line (a unit eigenvector) would be scaled to point on the line that is a distance of $4+\sqrt{72}$ from the origin. This is shown in Figure 1.

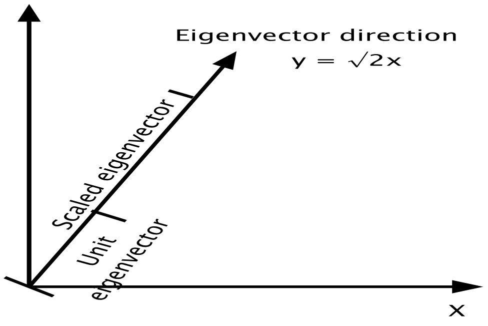


**FIGURE 1. The effect of a matrix on an eigenvector is to scale it, preserving its direction**

To obtain the other eigenvectors, we substitute the value $\lambda = 4 - \sqrt{72}$ in Equation 16 to get:

$$(4 - 4 + \sqrt{72})x_1 + 6x_2 = 0$$
$$12x_1 + (4 - 4 + \sqrt{72})x_2 = 0$$

$$\sqrt{72}x_1 + 6x_2 = 0$$
$$12x_1 + \sqrt{72}x_2 = 0$$

This gives us the parametrized eigenvector solution , $\begin{bmatrix} x_1 \\ -\sqrt{2}x_1 \end{bmatrix}$, which can be represented as $a\begin{bmatrix} 1 \\ -\sqrt{2} \end{bmatrix}$, where $a$ is an arbitrary scalar.

[]

### 3.5.3 Computing the eigenvalues of a matrix

We can compute the eigenvalues and eigenvectors of a matrix by generalizing the method of the previous example. Consider a matrix $A$ with an eigenvector $x$ and a corresponding eigenvalue $\lambda$. We know, by definition, that

$$Ax = \lambda x$$

We rewrite this as:

$$(A - \lambda I)x = \mathbf{0} \qquad \text{(EQ 17)}$$

This is a homogeneous system of equations, so from Section 3.4.5 on page 76 it has non-trivial solutions only if the determinant of the coefficient matrix is zero:

$$|A - \lambda I| = 0 \qquad \text{(EQ 18)}$$

This determinant is called the *characteristic determinant* of the matrix and Equation 18 is called its *characteristic equation*. Expanding the determinant will result, in general, in obtaining a polynomial of the *n*th degree in $\lambda$, which is the *characteristic polynomial* of the matrix. As we have seen, the eigenvalues of the matrix are the roots of the characteristic polynomial. This important result allows us to compute the eigenvalues of any matrix. Note also that the value of the characteristic determinant of a matrix does not change if the matrix is transposed. Hence, the eigenvalues of a matrix and its transpose are identical.

In general, the roots of a polynomial of degree *n* can be real or complex. Moreover, the fundamental theorem of algebra tells us that there is at least one root, and at most *n* distinct roots. Therefore, a square matrix of degree *n* has between one and *n* distinct eigenvalues, some of which may be complex (and will form complex conjugate pairs). Moreover, some eigenvalues may be repeated. Each eigenvalue corresponds to a family of eigenvectors that are parametrized by at least one free variable.

The set of eigenvalues of a matrix are called its *spectrum.* The largest eigenvalue by magnitude is called the *principal eigenvalue* or the *spectral radius* of the matrix. Each eigenvalue corresponds to a family of eigenvectors. It can be shown that the set of eigenvectors corresponding to a set of *distinct* eigenvalues are always linearly independent of each other. The set of all vectors that can be expressed as a linear combination of the eigenvectors is called the *eigenspace* of the matrix.

**Example 17: (Complex eigenvalues)**

Let us compute eigenvalues for the matrix $A = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$. We do so by setting the characteristic determinant to 0:

$$\begin{vmatrix} -\lambda & 1 \\ -1 & -\lambda \end{vmatrix} = 0$$

This gives us the characteristic equation:

$$\lambda^2 + 1 = 0$$

so that $\lambda = \pm i$. This is the spectrum of the matrix and the spectral radius is 1.

We find the eigenvector corresponding to $\lambda = i$ by setting

$$\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = i \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$x_2 = ix_1$$
$$-x_1 = ix_2$$

which is a set of equations with rank 1 (that is, the second equation just restates the first one). One possible vector that satisfies this is $\begin{bmatrix} 1 \\ i \end{bmatrix}$. To check this, note that $\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ i \end{bmatrix} = \begin{bmatrix} i \\ -1 \end{bmatrix} = i \begin{bmatrix} 1 \\ i \end{bmatrix}$. The eigenvector corresponding to the eigenvalue -*i* can be similarly found to be $\begin{bmatrix} x_1 \\ -ix_1 \end{bmatrix}$.

Because both eigenvector families are complex, the matrix never leaves the direction of a real-valued vector unchanged. Indeed, the matrix corresponds to a rotation by 90 degrees, so this is expected. What is unexpected that the rotation matrix does leave the 'direction' of a complex-valued vector unchanged, which has no obvious intuitive explanation.

[]

**Example 18: (Eigenvalues of a diagonal matrix)**

Consider a diagonal matrix, such as $A = \begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{bmatrix}$. What are its eigenvalues?

The characteristic equation is

$$\begin{vmatrix} a-\lambda & 0 & 0 \\ 0 & b-\lambda & 0 \\ 0 & 0 & c-\lambda \end{vmatrix} = 0$$

A little work shows that this reduces to the equation:

$$(a-\lambda)(b-\lambda)(c-\lambda) = 0$$

which shows that the eigenvalues are simply the diagonal elements *a*, *b*, and *c*. This generalizes: we can read off the eigenvalues as the diagonal elements of any diagonal matrix.

[]

### 3.5.4 Why are eigenvalues important?

The eigenvalues of a matrix become important when we consider the *repeated* application of a transformation on an input vector. Suppose we represent the state of a system by a vector. Suppose further that we can represent the transformation of the state in one time step as being equivalent to the application of a state transformation operator (and the equivalent matrix) on the state vector. Then, the 'steady-state' or eventual state of the system can be obtained by the repeated application of the transformation matrix on the initial vector. It turns out that the eigenvalues of the transformation matrix can be used to characterize the steady state.

To see this, first consider the repeated application of a matrix $A$ to its eigenvector $x$ corresponding to an eigenvalue $\lambda$. By definition, a single application of $A$ to $x = Ax = \lambda x$. Applying $A$ $n$ times therefore reduces to computing $\lambda^n x$, which is far simpler!

Now, consider an initial state vector $v$ that can be represented as a linear combination of two eigenvectors of $A$, say $x_1$, and $x_2$ like so:

$$v = c_1 x_1 + c_2 x_2$$

Suppose these eigenvectors correspond to the eigenvalues $\lambda_1$ and $\lambda_2$. Then, $A^n v$ can be found as follows:

$$A^n v = A^n(c_1 x_1 + c_2 x_2) = c_1 A^n x_1 + c_2 A^n x_2 = c_1 \lambda_1^n x_1 + c_2 \lambda_2^n x_2$$

We see that the repeated application of $A$ on $v$, which is a complex operation, is replaced by the far simpler operation of raising a scalar to the $n$th power.

This intuition is easily generalized. If we can represent the initial vector as the linear combination of the eigenvectors of a matrix, then the repeated application of the matrix can be found with little effort, as the next example shows.

**Example 19: (Computing $A^n x$ using eigenvectors)**

From the previous example, we know that the matrix $A = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$ has two eigenvalues *i,* and *-i*, with corresponding unit eigenvectors $\begin{bmatrix} 1 \\ i \end{bmatrix}$ and $\begin{bmatrix} 1 \\ -i \end{bmatrix}$. Consider vector $\begin{bmatrix} 10 \\ 0 \end{bmatrix}$. We can represent this as $5^* \begin{bmatrix} 1 \\ i \end{bmatrix} + 5^* \begin{bmatrix} 1 \\ -i \end{bmatrix}$. Therefore, $\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}^{100} \begin{bmatrix} 10 \\ 0 \end{bmatrix}$ (which is computationally complex) reduces to computing $5^* i^{100} {}^* \begin{bmatrix} 1 \\ i \end{bmatrix} + 5^* i^{100} {}^* \begin{bmatrix} 1 \\ -i \end{bmatrix} = 5^* \begin{bmatrix} 1 \\ i \end{bmatrix} + 5^* \begin{bmatrix} 1 \\ -i \end{bmatrix} = \begin{bmatrix} 10 \\ 0 \end{bmatrix}$. Geometrically, this makes sense, because applying $A$ once corresponds to a 90 degree rotation, so applying it for any multiple of four times will leave the initial vector unchanged.

[]

This method is useful when the initial vector can be written as a linear combination of the eigenvectors. In this context it is useful to know the following fact: the eigenvectors corresponding to a set of *distinct* eigenvalues form a linearly independent set. Any set of $n$ linearly independent complex vectors form a basis for the complex hyperspace $C^n$. Hence, if a square matrix of order $n$ has $n$ distinct eigenvalues, then we can express any initial vector as a linear combination of its eigenvectors.

What if the initial vector cannot be written as a linear combination of the eigenvectors? That is, what if the initial vector is not in the eigenspace of the matrix. It turns out that a somewhat more complex computation involving the eigenvalues allows us to compute $A^n x$. The details are, however, beyond the scope of this introduction.

### 3.5.5 The role of the principal eigenvalue

Consider a matrix $A$ that has a set of $m$ eigenvalues $\lambda_i$, $i = 1,2, ..., m$ with corresponding unit eigenvectors $x_i$. Suppose that we choose a vector $v$ in the eigenspace of the matrix such that it is expressed as a linear combination of *all* the eigenvectors:

$$v = \sum_{i=1}^{m} c_i x_i$$

Then, $n$ applications of the matrix to this vector results in the vector:

$$\sum_{i=1}^{m} c_i \lambda_i^n x_i$$

As $n \to \infty$, this sum is dominated by the eigenvalue that has the largest magnitude, which is called the *principal* or *dominant* eigenvalue (if more than one eigenvalue has the same magnitude, they are both considered to be the principal eigenvalues). To first approximation, we can ignore all the other eigenvalues in computing the limit.

**Example 20: (Principal eigenvalue)**

Consider the matrix $A = \begin{bmatrix} 4 & 6 \\ 12 & 4 \end{bmatrix}$. Recall that it has two eigenvalues, $\lambda = 4 \pm \sqrt{72}$, which evaluate to 12.48 and -4.48. The unit eigenvectors of this matrix are $\begin{bmatrix} 1 \\ \sqrt{2} \end{bmatrix}$ and $\begin{bmatrix} 1 \\ \frac{-1}{\sqrt{2}} \end{bmatrix}$. Suppose we start with an initial vector $\begin{bmatrix} 0 \\ 3\sqrt{2} \end{bmatrix}$ which we can express as 2* $\begin{bmatrix} 1 \\ \sqrt{2} \end{bmatrix}$ - 2* $\begin{bmatrix} 1 \\ \frac{-1}{\sqrt{2}} \end{bmatrix}$. Then, $A^{10} \begin{bmatrix} 0 \\ 3\sqrt{2} \end{bmatrix} = 2*12.48^{10}* \begin{bmatrix} 1 \\ \sqrt{2} \end{bmatrix} - 2*(-4.48)^{10}* \begin{bmatrix} 1 \\ \frac{-1}{\sqrt{2}} \end{bmatrix}$. This evaluates to $\begin{bmatrix} 1.83*10^{11} \\ 2.59*10^{11} \end{bmatrix}$. If we ignore the second term (i.e, the contribution due to the eigenvalue -4.48), then the resulting value changes only in the third decimal place of precision (check this!). It is clear that the dominant eigenvalue is the one that matters.

[]

### 3.5.6 Finding eigenvalues and eigenvectors

Computing the eigenvalues and the eigenvectors of a matrix is important in practice. Here are some simple facts that help identify the nature of the eigenvalues of a matrix.

- If a matrix is square and diagonal, then its eigenvalues are its diagonal elements.
- If a matrix is square and symmetric (that it, $A^T = A$), then its eigenvalues are real.
- *Gerschgorin's 'circle' theorem* states that all the eigenvalues of a complex matrix lie in the set of disks (on the complex plane) centered on the elements of the diagonal, with a radius equal to the sum of the magnitudes of the off-diagonal elements. Intuitively, if the off-diagonal elements are 'not too large,' then the eigenvalues of the matrix are its diagonal elements.

**Example 21: (Finding eigenvalues)**

The matrix $A = \begin{bmatrix} 9.1 & 0.8 & 0.3 \\ 0.8 & 5.8 & 0.2 \\ 0.3 & 0.2 & 6.5 \end{bmatrix}$ is symmetric. Hence, its eigenvalues are real.

It has three Gerschgorin disks: (1) Center 9.1 + 0*i*, radius 1.1 (2) Center 5.8 + 0*i* and radius 1.0 (3) Center 6.5+0*i* and radius 0.5. Because the eigenvalues are real, they must lie in one of three intervals: [8, 10.2], [4.8, 6.8], and [6, 7]. The second interval contains the third, so we know that the eigenvalues lie in one of the first two intervals.

[]

It is possible to approximately compute the dominant[2] eigenvalue of a matrix using the *power* method. In this technique, we start with an arbitrary initial vector $x_0$ and repeatedly apply $A$ to it. At each step, we compute the *Rayleigh ratio* $(x^T_k Ax_k)/(x^T_k x_k) = (x^T_k x_{k+1})/(x^T_k x_k)$, which will converge towards the dominant eigenvalue of the matrix. The idea is that applying $A$ to any vector will scale it in multiple dimensions, but the dominant eigenvalue will dominate the scaling effect. Repeatedly applying $A$ magnifies the contribution of the dominant eigenvalue, exposing it.

**Example 22: (Power method to compute the dominant eigenvalue)**

Let us use the power method to compute the dominant eigenvalue of the matrix $\begin{bmatrix} 4 & 6 \\ 12 & 4 \end{bmatrix}$. We start with the initial vector $x_0$ = $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ . Applying the matrix once, we get $x_1 = \begin{bmatrix} 10 \\ 16 \end{bmatrix}$. The ratio evaluates to( [1 1] * $\begin{bmatrix} 10 \\ 16 \end{bmatrix}$)/([ 1 1] * $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ ) = 26/2 = 13. Repeating, we get $x_2 = \begin{bmatrix} 136 \\ 184 \end{bmatrix}$ and the ratio evaluates to 4304/356 = 12.08. After one more iteration, we get $x_3 = \begin{bmatrix} 1648 \\ 2368 \end{bmatrix}$ , and the ratio evaluates to 659840/52352 = 12.60. Recall that the dominant eigenvalue is 12.48, and we have reached within 1% of it in only three iterations!

[]

It can be shown the speed of convergence of this method depends on the 'gap' between the dominant and second-largest eigenvalue. The bigger the gap, the faster the convergence. Intuitively, if the second-largest eigenvalue is close in magnitude to the dominant eigenvalue, then its scaling effects do not die down easily, requiring more iterations.

The power method can also be used to find the dominant *eigenvector.* To do so, after each iteration, the vector $x_i$ must be *scaled*. That is, we set its largest element to 1 by dividing each element by the largest element, as demonstrated next.

**Example 23: (Power method for computing the dominant eigenvector)**

Compute the dominant eigenvector of the matrix $\begin{bmatrix} 4 & 6 \\ 12 & 4 \end{bmatrix}$ using the power method.

We already know that $x_1 = \begin{bmatrix} 10 \\ 16 \end{bmatrix}$. We rescale it by dividing each element by 16 to get the vector $\begin{bmatrix} 0.625 \\ 1 \end{bmatrix}$ . Using this as the new value of $x_1$, we get $x_2 = \begin{bmatrix} 4 & 6 \\ 12 & 4 \end{bmatrix} * \begin{bmatrix} 0.625 \\ 1 \end{bmatrix} = \begin{bmatrix} 8.5 \\ 11.5 \end{bmatrix}$ . We rescale this to get $x_2 = \begin{bmatrix} 0.739 \\ 1 \end{bmatrix}$ . We compute $x_3 = \begin{bmatrix} 4 & 6 \\ 12 & 4 \end{bmatrix} * \begin{bmatrix} 0.739 \\ 1 \end{bmatrix} = \begin{bmatrix} 8.956 \\ 12.868 \end{bmatrix}$ which is rescaled to $\begin{bmatrix} 0.696 \\ 1 \end{bmatrix}$ . Recall that the eigenvector for this eigenvalue is exactly $\begin{bmatrix} 1 \\ \sqrt{2} \end{bmatrix}$ , which we scale to $\begin{bmatrix} 0.707 \\ 1 \end{bmatrix}$. With just three iterations, we are within 1.5% of the final value!

[]

[2] Usually, but not necessarily the dominant eigenvalue, it turns out, though in nearly all practical cases, this is the one that will be found.

### 3.5.7 Similarity and diagonalization

Two matrices are said to be *similar* if they have the same set of eigenvalues. In some cases, given a matrix *A*, it is useful to be able to compute a similar *diagonal* matrix *D* (we show a use case below).

A sufficient, but not necessary, condition for a matrix of size *n* to be diagonalizable is that it has *n* distinct eigenvalues. In this case, let *X* denote a matrix whose columns are the eigenvectors of *A*. Then, it can be shown (through expansion of the underlying terms) that the matrix

$$D = X^{-1}AX \qquad \textbf{(EQ 19)}$$

is a diagonal matrix whose diagonal elements are the eigenvalues of *A*.

Knowing the diagonalized version of a matrix is useful in computing its $m^{\text{th}}$ power. From Equation 19, note that

$$D^2 = (X^{-1}AX)(X^{-1}AX) = X^{-1}A^2X$$

A simple induction shows that

$$D^m = X^{-1}A^mX$$

so that

$$A^m = XD^mX^{-1}$$

But the right hand side is easily computed, since *D* is diagonal. Hence, we can easily compute $A^m$ as the next example shows.

**Example 24: (Diagonalization)**

Consider the matrix $A = \begin{bmatrix} 4 & 6 \\ 12 & 4 \end{bmatrix}$. Recall that it has two eigenvalues, $\lambda = 4 \pm 6\sqrt{2}$ corresponding to the eigenvectors $\begin{bmatrix} 1 \\ \sqrt{2} \end{bmatrix}$ and $\begin{bmatrix} 1 \\ -\sqrt{2} \end{bmatrix}$. We write out the matrix *X* as $\begin{bmatrix} 1 & 1 \\ \sqrt{2} & -\sqrt{2} \end{bmatrix}$. From Equation 14 we find that $X^{-1} = \dfrac{1}{(-\sqrt{2}-\sqrt{2})}\begin{bmatrix} -\sqrt{2} & -1 \\ -\sqrt{2} & 1 \end{bmatrix}$. Therefore, we diagonalize *A* as

$$\frac{1}{-2\sqrt{2}}\begin{bmatrix} -\sqrt{2} & -1 \\ -\sqrt{2} & 1 \end{bmatrix}\begin{bmatrix} 4 & 6 \\ 12 & 4 \end{bmatrix}\begin{bmatrix} 1 & 1 \\ \sqrt{2} & -\sqrt{2} \end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix} \sqrt{2} & 1 \\ \sqrt{2} & -1 \end{bmatrix}\begin{bmatrix} 2 & 3 \\ 6 & 2 \end{bmatrix}\begin{bmatrix} 1 & 1 \\ \sqrt{2} & -\sqrt{2} \end{bmatrix} = \begin{bmatrix} 4+6\sqrt{2} & 0 \\ 0 & 4-6\sqrt{2} \end{bmatrix}$$

Note that the diagonal elements of *A* are its eigenvalues.

To compute $A^5$, we compute

$$\begin{bmatrix} 1 & 1 \\ \sqrt{2} & -\sqrt{2} \end{bmatrix}\begin{bmatrix} 4+6\sqrt{2} & 0 \\ 0 & 4-6\sqrt{2} \end{bmatrix}^5 \frac{1}{-2\sqrt{2}}\begin{bmatrix} -\sqrt{2} & -1 \\ -\sqrt{2} & 1 \end{bmatrix}$$

Due to diagonalization, the matrix power computation reduces to computing the exponential of a scalar value. After simplification (and maintaining 10 digits of precision in the calculations), this reduces to

$$\begin{bmatrix} 150783.99 & 107903.99 \\ 215807.99 & 150783.99 \end{bmatrix}$$

which is within rounding error of the true value of

$$\begin{bmatrix} 150784 & 107904 \\ 215808 & 150784 \end{bmatrix}$$

[]

Note that this technique allows us to compute the effect of $A^m$ on *all* possible vectors, rather than only on vectors expressed as a linear combination of eigenvectors, as we had required in Section 3.5.4 on page 82.

## *3.6 Stochastic matrices*

We now turn our attention to a special type of matrix called a *stochastic matrix*. A *right stochastic matrix* is a square matrix whose elements are non-negative reals and each of whose rows sums to 1. A *left stochastic matrix* is a square matrix whose elements are non-negative reals and each of whose *columns* sums to 1. Stochastic matrices are also called *Markov matrices*. Unless otherwise specified, when we refer to a 'stochastic matrix,' we will refer to a *right* stochastic matrix.

Stochastic matrices are important in the context of computer networking because each row of such a matrix *A* corresponds to the state of a finite state machine (or Markov chain) representing a networking protocol or a buffer in a router. Each element $a_{ij}$ can be viewed as the probability of entering state *j* from state *i*. The summation criterion expresses the fact that the result of a transition from a state is to either remain in the same state or to go to some other state. Stochastic matrices arise frequently in the study of Markov chains, stochastic processes, and in queueing theory.

**Example 25: (Stochastic matrix)**

The matrix $A = \begin{bmatrix} 0.25 & 0.5 & 0.25 \\ 0.1 & 0.9 & 0 \\ 0 & 0 & 1.0 \end{bmatrix}$ is a stochastic matrix because it is square, the elements are non-negative reals, and each row sums to 1. We interpret row 1 to mean that if the system is in state 1, then after one transition, it remains in state 1 with probability 0.25, goes to state 2 with probability 0.5, and goes to state 3 with probability 0.25. Note that if the system enters state 3, then it can never leave that state (Why?). We call such a state an *absorbing* state.

[]

### 3.6.1 Computing state transitions using a stochastic matrix

Consider a $n \times n$ stochastic matrix *A* and a column vector *p* having dimension *n* with non-negative real elements such that its elements sum to 1. We think of the *i*th element of *p* as representing the probability of being in state *i* at some point in time. Then $p^{'}=A^Tp$ is a vector whose *i*th element is the probability of being in state *i* after one transition (note the pre-multiplication not by *A*, but by its transpose). This is because the *i*th element of *p'* is given by $p'_i = \sum_{k=1}^{n} p_k a_{ki}$, which is total probability of being in state *i* after the transition, conditioning on the probability of being in each prior state *k*.

**Example 26: (State transitions)**

Continuing with the stochastic matrix from Example 25, suppose that we start with the system is in state 1. What is the probability of being in state 1 after one and two transitions?

The initial state vector is $p$= [1.0 0 0]$^T$. After one transition, the state vector is given by $p= A^Tp$ =

$$\begin{bmatrix}0.25 & 0.1 & 0\\ 0.5 & 0.9 & 0\\ 0.25 & 0 & 1.0\end{bmatrix}\begin{bmatrix}1\\0\\0\end{bmatrix} = \begin{bmatrix}0.25\\0.5\\0.25\end{bmatrix}$$

and after two transitions $p= A^T(A^Tp) = (A^T)^2p$ =

$$\begin{bmatrix}0.1125 & 0.115 & 0.3125\\ 0.575 & 0.86 & 0.025\\ 0.3125 & 0 & 1.0\end{bmatrix}\begin{bmatrix}1\\0\\0\end{bmatrix} = \begin{bmatrix}0.1125\\0.575\\0.3125\end{bmatrix}$$

Thus, after two transitions, the system is in state 1 with probability 0.1125. Note that this probability is larger than the simple probability of staying in state 1 (which is just 0.25 * 0.25 = 0.0625), because it takes into account the probability of transitioning from state 1 to state 2 and then back from state 2 to state 1, which has an additional probability of 0.5*0.1 = 0.05.

[]

As this example shows, if $A$ is a stochastic matrix, then the $[i,j]^{\text{th}}$ element of $(A^T)^2$ represents the probability of going from state $i$ to state $j$ in two steps. Generalizing, the probability of going from state $i$ to state $j$ in $k$ steps is given by $(A^T)^k$.

### 3.6.2 Eigenvalues of a stochastic matrix

We now present two important results concerning stochastic matrices.

First, *every* stochastic matrix has an eigenvalue of 1. To prove this, consider the $n \times n$ stochastic matrix $A$ and column vector $x$ = [1 1 ... 1]$^T$. Then, $Ax = x$, because each element of $Ax$ multiplies 1 with the sum of a row of $A$, which, by definition, adds to 1. Because a matrix and its transpose have the same eigenvalues, the transpose of a stochastic matrix also has an eigenvalue of 1. However, the eigenvector of the transposed matrix corresponding to this eigenvalue need not be (and rarely is) the **1** vector.

Second, every (possibly complex) eigenvalue of a stochastic matrix must have a magnitude smaller than 1. To prove this, consider some diagonal element $a_{jj}$. Suppose this element takes the value $x$. Then, by definition of a stochastic matrix, it must be the case that the sum of the off-diagonal elements is 1-$x$. From Gerschgorin's circle theorem, we know that all the eigenvalues lie within a circle in the complex plane centered at $x$ with radius 1-$x$. The largest magnitude eigenvalue will be a point on this circle (see Figure 2). Although the truth of the proposition is now evident by inspection, we now formally prove the result.

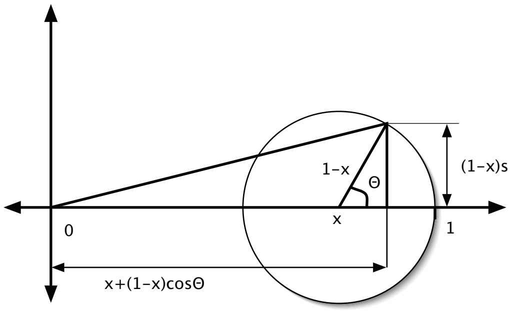


FIGURE 2. **Largest possible eigenvalue of a stochastic matrix**

Suppose that this point subtends an angle of $\theta$. Then, its coordinates are ($x$+(1-$x$) cos$\theta$, (1-$x$)sin$\theta$). Therefore, its magnitude is $((x+(1-x)\cos\theta)^2+((1-x)\sin\theta)^2)^{1/2}$, which simplifies to $(x^2 + (1-x)^2+2x(1-x)\cos\theta)^{1/2}$. This quantity is maximized when $\theta$=0 so that we merely have to maximize the quantity $x^2 + (1-x)^2+2x(1-x)$. Taking the first derivative with respect to $x$ and setting it to zero shows that this expression reaches its maximum of 1 independent of the value of $x$. So, we can pick $x$ to be a convenient value, such as 1. Substituting $\theta$=0 and $x$ = 1 into $((x+(1-x)\cos\theta)^2+((1-x)\sin\theta)^2)^{1/2}$, we find that the magnitude of the maximum eigenvalue is 1.

It can also be shown, though the proof is beyond the scope of this text, that only *one* of the eigenvalues of a stochastic matrix is 1. This means that only one eigenvalue of its transpose is 1. Therefore, we can use the power method to find the dominant eigenvector $p$ corresponding to this dominant eigenvalue. This eigenvector is interesting because the probability of being in any state $i$ remains *unchanged* after a state transition (which, recall, corresponding to a multiplication of the state vector by $A^T$ - not $A$). Therefore, we call $p$ the *stationary probability* distribution corresponding to $A^T$ and it is a *fixed point* corresponding to $A^T$, in that the application of $A^T$ does not change this point. In our study of queueing theory, we will study the conditions on the matrix $A^T$ which guarantee the existence of such a stationary probability, and guarantee further that the stationary probability distribution is reached independent of the initial value of $p$. These are the conditions under which the matrix $A^T$ is said to be *ergodic*.

**Example 27: (Google page rank algorithm)**

The power technique of finding the dominant eigenvector of a stochastic matrix can be used to rank a set of web pages. More precisely, given a set of web pages, we would like to identify certain pages as being more important than others. A page can be considered to be important using the recursive definition that (a) many other pages point to it and (b) the other pages are also important.

The importance of a page can be quantified according the actions of a ‘random web surfer’ who goes from web page $i$ to a linked web page $j$ with probability $a_{ij}$. If a page is ‘important,’ then a random web surfer will be led to that page more often than to other less-important pages. That is, we consider a population of a large number of surfers, then a larger fraction of web surfers will be at a more important page, compared to a less important page. Treating the ratio of the number of web surfers at a page to the total number of surfers as approximating a probability, we see that the importance of a page is just the stationary probability of being at that page.

To make matters more precise, let the matrix $A$ represent the set of all possible transition probabilities. If the probability of the surfer being at page $i$ at some point is $p_i$ then the probability that the surfer is at page $i$ after one time step is $A^Tp$. The dominant eigenvector of $A^T$ is then the ‘steady state’ probability of a surfer being at page $i$. Given that $A$ is a stochastic matrix, we know that this dominant eigenvector exists, and that it can be found by the power method.

What remains is to estimate the quantities $a_{ij.}$ Suppose page $i$ has links to $k$ pages. Then, we set $a_{ij} = 1/k$ for each page $j$ to which it has a link, and set $a_{ij}$ = 0 for all other $j$. This models a surfer going from a page uniformly randomly to one of its linked pages. What if a page has no links? Or if two pages link only to each other? These issues can be approximately modelled by assuming that, with constant probability, the surfer ‘teleports’ to a randomly chosen page. That is, if there is a link from page $i$ to page $j$, $a_{ij} = \alpha/n + (1-\alpha)/k$, where $\alpha$ is a control parameter; otherwise $a_{ij} = \alpha/n$. It can be easily shown that these modified $a_{ij}$s form a stochastic matrix, so that we can extract the dominant eigenvalue, and thus the page rank, using the power method. A slightly modified version of this algorithm is the publicly known algorithm used by Google.

[]

## 3.7 Exercises

**1 Transpose**

Compute the transpose of $\begin{bmatrix} 4 & 0 & -3 \\ 7 & 82 & 12 \\ 3 & -2 & 2 \end{bmatrix}$ .

**2 Matrix multiplications**

Find the product of the matrices $\begin{bmatrix} 10 & -4 & 12 \\ -5 & 3 & 9 \\ 8 & 0 & -4 \end{bmatrix}$ and $\begin{bmatrix} -4 & 5 & 3 \\ 7 & -2 & 9 \\ 2 & 5 & -4 \end{bmatrix}$ .

**3 Exponentiation**

Prove that if $A$ is a diagonal matrix the $(i,i)^{th}$ element of the $k$th power of $A$ is $a_{ii}^{k}$.

**4 Linear combination of scalars**

Compute the linear combination of the scalars 10, 5, 2, -4 with weights 0.5, 0.4, 0.25, 0.25.

**5 Linear combination of vectors**

Compute the linear combination of the vectors [1 2 8 5], [3 7 3 1], [7 2 1 9], [2 6 3 4] with weights 0.5, 0.4, 0.25, 0.25.

**6 Linear independence and rank**

Are the three vectors:

$x_1$ = [12 2 -4]

$x_2$ = [2 2 -24]

$x_3$ = [2.5 0 5]

independent? Determine this from the rank of the corresponding coefficient matrix.

**7 Basis and dimension**

Give two possible bases for the three vectors in Exercise 6. What is the dimension of the vector space generated by these bases?

**8 Gaussian elimination**

Use row operations and Gaussian elimination to solve the system given by $\begin{bmatrix} 6 & 4 & -8 & 5 \\ -8 & 2 & 4 & -2 \\ 10 & 0 & 4 & 1 \end{bmatrix}$ .

**9 Rank**

Prove that the rank of an $n \times n$ non-zero diagonal matrix is $n$.

**10 Determinant**

Compute the determinant of the matrix $\begin{bmatrix} 4 & 0 & -3 \\ 7 & 8 & 12 \\ 3 & -2 & 2 \end{bmatrix}$ .

**11 Inverse**

Compute the inverse of the matrix $\begin{bmatrix} 4 & 0 & -3 \\ 7 & 8 & 12 \\ 3 & -2 & 2 \end{bmatrix}$ .

### 12 Matrix as a transformation

Using the fact that sin(A+B) = sin(A)cos(B) + cos(A)sin(B) and cos(A+B) = cos(A)cos(B) - sin(A)sin(B), compute the matrix that corresponds to the rotation of a vector joining the origin to the point (*x, y*) by an angle *p*.

### 13 Composing transformations

Compute the composition of a rotation of *t* degrees followed by a rotation of *p* degrees.

### 14 Eigenvalues and eigenvectors

Compute the eigenvalues and corresponding eigenvectors of the matrix $\begin{bmatrix} 1 & 9 \\ 4 & 1 \end{bmatrix}$.

### 15 Computing $A^n x$

Find the value of $\begin{bmatrix} 1 & 9 \\ 4 & 1 \end{bmatrix}^5 \begin{bmatrix} 8 \\ 0 \end{bmatrix}$.

### 16 Finding eigenvalues

Bound the interval(s) in which the eigenvalues of the matrix $\begin{bmatrix} 4 & 1 & 0.5 \\ 1 & 6 & 0.3 \\ 0.5 & 0.3 & 5 \end{bmatrix}$ lie.

### 17 Power method

Use the power method to compute the dominant eigenvalue and corresponding eigenvector of the matrix $\begin{bmatrix} 1 & 9 \\ 4 & 1 \end{bmatrix}$.

Iterate four times.

### 18 Diagonlization

What is the diagonal matrix similar to $\begin{bmatrix} 1 & 9 \\ 4 & 1 \end{bmatrix}$ ?

### 19 Stochastic matrix

Is the matrix $\begin{bmatrix} 0.1 & 0.8 & 0.3 \\ 0.5 & 0.1 & 0.4 \\ 0.4 & 0.1 & 0.3 \end{bmatrix}$ stochastic?

### 20 State transitions

Consider a system described by the stochastic matrix $\begin{bmatrix} 0.25 & 0.5 & 0.25 \\ 0.1 & 0.9 & 0 \\ 0 & 0 & 1.0 \end{bmatrix}$. Let the *i*th row of this matrix correspond to state *i*. If the initial state is known to be state 1 with probability 0.5 and state 2 with probability 0.5, compute the probability of being in these two states after two time steps.

CHAPTER 4

# *A Tourist's Guide to Optimization*

This chapter presents an overview of optimization: a set of mathematical tools that can be used to model and improve the performance of a computer system.

## *4.1 System modelling for optimization*

A necessary prerequisite to the use of optimization techniques is to mathematically model a system in terms of:

1. The *fixed parameters*. These are aspects of the system that cannot be changed, and therefore, from the perspective of the model, are constants.
2. The *control parameters*. These are the "tuning knobs" or settings that can be chosen to optimize the behavior of the system. Control parameters are typically constrained to lie within some range. A set of control parameters where each parameter is within its valid range is called a *feasible* set of control parameters.
3. *Input variables*. These are external and uncontrollable inputs to the system. For a particular instance of a general model, an input variable can be considered to be a fixed parameter.
4. *Output variables* (or *performance metrics*). These are the observable outputs of the system. Some output variables are chosen as performance metrics (variables that quantify the performance of the system).
5. The *transfer function*. This maps from input, fixed, and control parameters to the output variables.

### 4.1.1 Optimization

Given this model, *optimization* is the process of choosing a feasible set of control parameters so that an *objective function O* defined over the output variables is optimized (either maximized or minimized). Note that the objective function can be defined directly as a function of the control parameters, the fixed parameters and the input variables by using the transfer function to rewrite each output variable in the objective function. Moreover, the fixed parameters and input variables can be considered to be system-defined constants. Therefore, the objective function is typically represented as a function whose variables are the control parameters and whose constants are the fixed parameters and input variables. We will use this notation from now on.

**Example 1 (Mathematical modelling of a problem):**

Consider a communication path from a source to a destination that has a capacity of 100 packets/sec. How fast should a source send to maximize its performance, if this performance is specified by the difference between the carried load and the drop rate?

We model the system as follows:

- *Fixed parameter*: capacity of the path, i.e 100 packets/s
- *Control parameter:* the source sending rate
- *Input variables*: none
- *Output variables*: carried load and drop rate
- *Objective function:* carried load - drop rate
- *Transfer function*:
  carried load = min{sending rate, 100} packets/sec
  drop rate = max(0, sending rate - 100) packets/sec

The objective function can be written in terms of the control parameters and fixed parameters as:

objective function = carried load - drop rate
= min(sending rate, 100) - max(0, sending rate-100) packets/sec

which is shown in Figure 1.The objective function is maximized when the sending rate is 100 packets/s.

[]

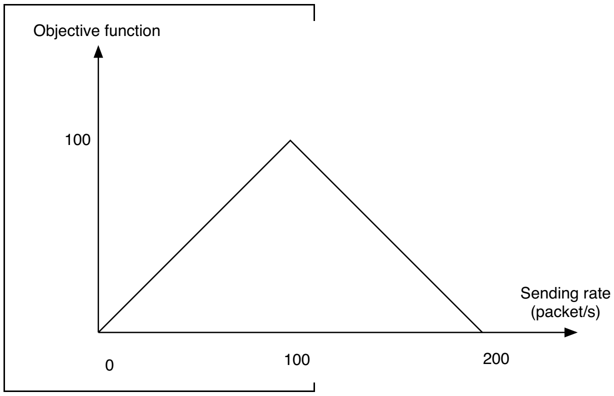


**FIGURE 1. The objective function for Example 1. The maximum occurs when the control variable is set to 100 packets/second.**

In this example, the system model could be derived with little effort. In practice, choosing an appropriate model is an art that is only learned with much experience. Moreover, we could graph the transfer function because the objective function depends only on one control parameter. What if we had hundreds of control parameters? In such cases, we cannot graph the system, and must resort to a more sophisticated mathematical analysis, as described in the remainder of this guide.

## *4.2 An example*

**Example 2 (Optimizing a system with two control parameters):**

We will ease our way into optimization through the following example. Consider a system whose objective function *O* can be expressed in terms of two control parameters $x_1$ and $x_2$ as:

$$O = 2x_1 - x_{2,} \text{ where} \quad \textbf{(EQ 1)}$$

$$x_1 + x_2 = 1 \text{ and} \quad \textbf{(EQ 2)}$$

$$x_1 \geq 0\ x_2 \geq 0 \quad \textbf{(EQ 3)}$$

Observe that $O$ increases when $x_1$ increases, and decreases when $x_2$ increases. So, to maximize $O$, we should set $x_2$ to 0, which implies that $x_1$ = 1, and $O$ = 2. We can express this geometrically as shown in Figure 2:

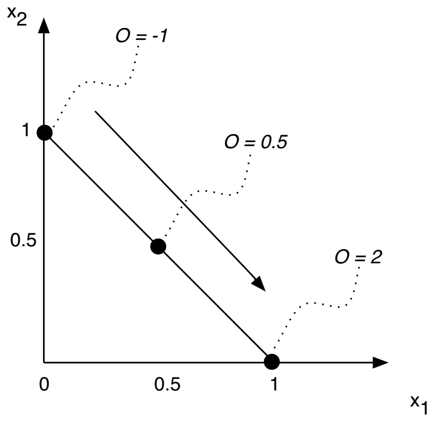


FIGURE 2. **Maximizing a function of two variables**

Note that the constraint on the $x_i$ s means that allowed values lie on a line defined by (0,1) and (1,0). At (0,1), $O$ is -1. As we move down the line, $x_1$ increases and $x_2$ simultaneously decreases, so that $O$ continuously increases, reaching its maximum value of 2 at (1,0).

**Example 3 (Optimizing a system with three variables):**

This example is easily generalized to three variables. Consider a system where:

$$O = 3x_1 - x_{2,} - x_3 \text{ and} \quad \textbf{(EQ 4)}$$

$$x_1 + x_2 + x_3 = 1 \text{ and} \quad \textbf{(EQ 5)}$$

$$x_1 \geq 0\ x_2 \geq 0\ x_3 \geq 0 \quad \textbf{(EQ 6)}$$

This is illustrated geometrically in Figure 3 below:

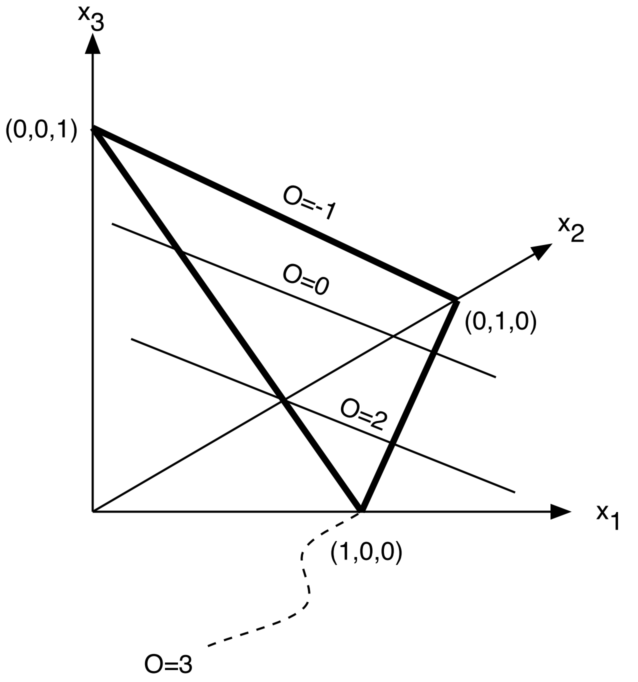


**FIGURE 3.** **Maximizing a function of three variables The constraint plane is shown with a bold outline.**

The constraints require the $x_i$s to lie on the subset of a plane defined by (0,0,1), (0,1,0), and (1,0,0) (we call this a *constraint plane*). Moreover, the constraints that the $x_i$s are positive imply that the solution lies within the solid defined by this plane and the three other constrain planes: $x_1 = 0$, $x_2 = 0$, and $x_3 = 0$.

Note that at (0,0,1), $O$ is -1. At every point on the line defined by (0,0,1) and (0,1,0) and also lying on the constraint plane $x_1 + x_2 + x_3 = 0$, $x_1$ is 0, and therefore $x_2 + x_3 = 1$. Consequently, $O$ is -1 on this entire line. If, however, we move anywhere on this constraint plane other than on this line, $x_1$ is non-zero, which means that $O > -1$ .

In Figure 3, consider the two lines on the constraint plane parallel to the bold line on the $x_2$ - $x_3$ plane defined by (0,0,1) and (0,1,0). These lines are defined by the intersection of the plane$x_1$=$C$ with the constraint plane. Both lines have a constant value for $x_1$ and at every point on the line the sum of $x_2$ and $x_3$ is a constant smaller than 1 (Why?). Therefore, along each such line, $O$ is also constant. In fact, there are an infinite number of lines like these on the constraint plane, called *isoquant* lines, where $O$ is constant.

Visually, if you imagine drawing a line overlaying the isoquant where $O$ is -1, and then moving this line towards the point (1,0,0) while keeping it parallel to the $x_2$ - $x_3$ plane, the value of $O$ rises monotonically and attains its maximum value when the sweeping line departs from the constraint plane at the vertex (1,0,0).

It is easy to see that if we take any two points A and B on the constraint plane, one of three conditions must hold. Either the value of $O$ is the same at both A and B (in which case they are on the same isoquant), or $O$ is greater at A, or $O$ is greater at B. Put another way, if we take any point A on the constraint plane and look in its neighbourhood for another point B also on the constraint plane, $O$ will have the same value at B as at A, or, if B is closer to (1,0,0), then $O$ will be larger at B, or if B is further away from (1,0,0) from A, then $O$ will have a larger value at A.

This suggests the following optimization procedure: start at a random point on the constraint plane and somehow generate a set of neighbours of this point. Then evaluate $O$ at those neighbours that are also on the constraint plane. With any luck, one of these *valid* neighbours will be closer to (1,0,0), and so will have a higher value of $O$. We move our attention to this point and repeat the process. When the process terminates, we will have reached (1,0,0) and no neighbour of this point would have a higher value of *O*. This finds the optimal solution.

In general, independent of the form of the objective and constraint functions, we can always define isoquants corresponding to sets of points that satisfy the constraints and that all have the same value of *O* (the points in these sets may not be contiguous). We find the optimal value of *O* by hopping from isoquant to isoquant such that *O* always increases. This is the basis of the optimization technique called *hill climbing* that we will return to later.

Now, we restrict our attention to linear systems, that is, systems where both the constraints and the objective function are linear. The systems in Examples 2 and 3 are both linear systems. For such systems, one can intuitively see that *O* is maximized (or minimized) at one of the vertices of the constraint plane. Why? Because, at any other point we can find a neighbour that lies on a better (respectively worse) isoquant. It is only at a vertex that we run out of neighbours[1]. Of course, in some cases, the isoquant can be parallel to one of the edges of the constraint plane. In this case, the *O* attains a minimum or maximum along an entire edge. If so, we can find the extremal value of *O* simply be examining *only the vertices of the constraint plane*. This fact is crucial in optimizing linear systems, which is the focus of the next sub-section.

## 4.3 Optimizing linear systems

We often wish to minimize an objective function *O* that is a linear combination of the control parameters $x_1, x_2,... x_\mathrm{n}$ and can therefore be expressed as:

$$O = c_1x_1 + c_2x_2 + \ldots + c_nx_n$$

Moreover, the $x_i$s are positive and constrained by linear constraints of the form:

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1$$

$$a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2$$

$$\ldots$$

$$\ldots$$

$$a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n = b_m$$

or, using matrix notation:

$$\boldsymbol{Ax} = \boldsymbol{b}$$

$$\boldsymbol{x} \geq \boldsymbol{0}$$

where **A** is an $m \times n$ matrix, and **x** and **b** are column vectors with *m* and *n* elements respectively.

Each equality corresponds to a hyperplane, which is a generalization of a plane to more than two dimensions. The constraints ensure that valid $x_i$s lie at the intersection of these hyperplanes.

Note that we can always transform an inequality of the form

[1] For general objective functions, we could run out of better points even within the constraint plane, so the optimal point may not lie at a vertex.

$$a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n \geq b_i$$

to an equality by introducing a new variable $s_i$ called the *surplus variable* such that

$$a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n - s_i = b_i$$

By treating the $s_i$ as a virtual control parameters, we can convert a constraint that has a greater-than inequality into the standard form (at the end, we ignore the value assigned to a surplus variable). Similarly, introducing a *slack* variable converts lesser-than inequalities to equalities. Therefore, any linear system of equal and unequal constraints can be transformed into the standard form. Once this is done, we can use *linear programming* to find the value of **x** that maximizes the objective function.

**Example 4 (Representing a linear program in standard form):**

Consider a company that has two network connections to the Internet through two providers (this is also called *multi-homing*). Suppose that the providers charge per-byte and provide different delays. For example, the lower-priced provider may guarantee that transit delays are under 50ms, and the higher-priced provider may guarantee a bound of 20ms. Suppose the company has two commonly used applications, A and B, that have different sensitivities to delay. Application A is more tolerant of delay than application B. Moreover, the applications, on average, generate a certain amount of traffic every day, all of which has to be carried by one of the two links. The company wants to allocate *all* the traffic from the two applications on the two links, maximizing their benefit while minimizing its payments to the link providers. Represent the problem in standard form.

The first step is to decide how to model the problem. We need to have variables that reflect the traffic sent by each application on each link. Let's call the lower-priced provider *l* and the higher priced provider *h*. Then we can denote the traffic sent by A on *l* as $x_{Al}$ and the traffic sent by A on *h* as $x_{Ah}$. We can similarly define $x_{Bl}$ and $x_{Bh}$. The traffic sent is non-negative, so we immediately have:

$$x_{Al} \geq 0 \; ; \; x_{Ah} \geq 0 \; ; \; x_{Bl} \geq 0 \; ; \; x_{Bh} \geq 0 \; ;$$

Moreover, if we denote the traffic sent each day by application A by *TA* and the traffic sent by B by *TB*, we have:

$$x_{Al} + x_{Ah} = TA \; ; \; x_{Bl} + x_{Bh} = TB$$

Suppose that the providers charge $c_l$ and $c_h$ monetary units per byte. Then, the cost to the company is:

$$x_{Al}c_l + x_{Bl}c_l + x_{Ah}c_h + x_{Bh}c_h = C$$

What is the benefit to the company? Suppose that application A gains a benefit of $b_{Al}$ per byte from sending traffic on link *l* and $b_{Ah}$ on link *h*. Using similar notation for the benefits to application B, the overall benefit (i.e., benefit -cost) that the company is trying to maximize (the objective function) is:

$$O = (b_{Al} - c_l)x_{Al} + (b_{Ah} - c_h)x_{Ah} + (b_{Bl} - c_l)x_{Bl} + (b_{Bh} - c_h)x_{Bh}$$

Thus, in standard form, the linear program is the objective function above, and the constraints on the variables expressed as:

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_{Al} \\ x_{Ah} \\ x_{Bl} \\ x_{Bh} \end{bmatrix} = \begin{bmatrix} TA \\ TB \end{bmatrix}$$

$$\begin{bmatrix} x_{Al} \\ x_{Ah} \\ x_{Bl} \\ x_{Bh} \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

[]

How should we optimize such a linear program-how can we find values of the $x_{ij}$ such that $O$ is maximized? Trying every possible value of **x** is an exponentially hard task, so we have to be cleverer than that. What we need is an algorithm that systematically chooses $x_i$s that maximize or minimize $O$.

## *4.4 Solving a linear system using simplex*

To solve a linear system in standard form, we draw on the intuition developed in Examples 2 and 3. Recall that in Example 3, the optimal value of $O$ was reached at one of the vertices of the constraint plane. In a general system, the constraint plane is replaced by a mathematical object called a *polytope* defined as a convex hyperspace bounded by a set of hyperplanes. For instance, it is the three-dimensional solid bounded by the constraint planes in Figure 3.

A polytope in more than three dimensions is difficult to imagine: for instance, the intersection of two four-dimensional hyperplanes is a three-dimensional solid! The important point about a polytope is that each of its vertices is defined by $n$ coordinates, which are the values assumed by the $x_i$s at that vertex. The optimal value of $O$ is achieved for the values of the $x_i$s corresponding to the optimal vertex.

How do we find the optimal vertex? We first find some vertex of the polytope (which, by definition satisfies the constraints of the system). This is easy to find: we set enough of the $x_i$s to 0 so that the resulting system is not underconstrained and solve the linear system using, for example, Gaussian elimination. The solution to the system is, by definition, a vertex of the solution polytope. We then move from this vertex to an adjoining vertex such that $O$ increases (in fact, we could move to the adjoining vertex that improves $O$ the most). We repeat this process until we find a vertex such that value of $O$ at that vertex is greater than its value at all of that vertex's neighbours, which is the optimal vertex. This algorithm, developed by G. Dantzig, is the famous *simplex* algorithm.

To carry out simplex in practice, we must ensure that the polytope is well-formed. This means we have to identify and eliminate incompatible constraints. Moreover, it is possible for a set of vertices to have the same exact value of $O$, which can lead to infinite loops in the simplex algorithm. We can eliminate these problems by slightly jittering the value of $O$ at these vertices and employing more sophisticated *anti-looping* algorithms.

From the perspective of a practitioner, all that needs to be done is to specify the objective function and the constraints to a program called a Linear Program Solver or LP Solver. CPLEX and CS2 are two examples of well-known LP Solvers. The program returns either the optimal value of the objective function and the vertex at which it is achieved or declares the system to be unsolvable due to incompatible constraints.

Today's LP Solvers can routinely solve systems with more than 100,000 variables and tens of thousands of constraints.

### 4.4.1 The complexity of LP

The simplex algorithm, though efficient in practice, can in the worst case take time exponential in the size of the input (i.e, the number of variables). Another LP solution algorithm, called the ellipse method, is guaranteed to terminate in $O(n^3)$ time, where $n$ is the size of the input. Finally, a recent approach called the *interior point method* finds the optimal vertex not by moving from vertex to vertex, but by using points interior to the polytope. Such methods are both fast and computationally efficient. << Do I need to say more on these?>>

## 4.5 Using linear programming

With an appropriate choice of variables, LP can be used to solve many standard optimization problems. As an example, we now consider how to solve the network flow problem using LP.

### 4.5.1 Network flow

Network flow models the flow of commodities in a network, such as the shipment of goods across a country by means of trucks, roads, and warehouses. Warehouses are considered to have unlimited space (at least for this simple version of the problem), and each road has a fixed capacity, presumably related to the number of lanes it has. We will assume the we are not limited by the number of trucks, either. The problem is to send as much as possible between a node, called a source, to the ultimate destination, called the sink.

We represent the road network by a graph, where a node corresponds to a warehouse and each edge to a road. A directed edge is associated with a real-valued capacity, which is the capacity of the road in that direction. We define two distinguished nodes, called the *source* which has no edges entering it, and a *sink* which has no edges leaving it. Our goal is to find the largest amount of traffic that can be sent from the source to the sink over the entire graph.

**Example 5 (Network flow):**

Consider the network flow graph in Figure 4. Here, the node *s* represents the source and has a total capacity of 11.6 leaving it. The sink, usually denoted by *t* has a capacity of 25.4 entering it. The maximum capacity from *s* to *t* can be no larger than 11.6, but may be smaller, depending on the intermediate paths.

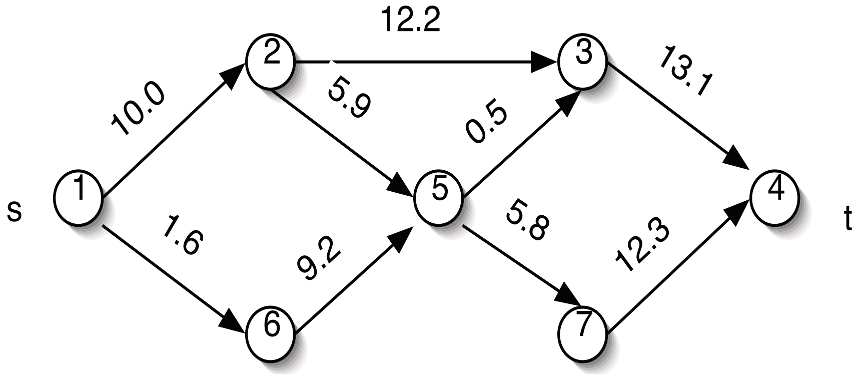


FIGURE 4. **Example of a network flow problem**

[]

We can compute the maximal flow that can be sustained on a network flow graph using linear programming. We denote the capacity of the link *ij* from *i* to *j* by $c_{ij}$ and the amount of traffic assigned to that link (as part of a flow from *s* to *t*) by $f_{ij}$. For example, in Figure 4, $c_{12}$ = 10.0 and we may end up assigning $f_{12}$ = 2.3 on it as part of the overall flow from *s* to *t*. We can then express two types of constraints on the $f_{ij}$s:

1. Capacity constraints: the flow on a link cannot exceed its capacity, that is, $f_{ij} \le c_{ij}$.
2. Conservation conditions: all the flow entering a node (other than the sink) must exit it; that is, for all nodes *j* other than *s* and *t*, $\sum_{\forall i | \exists ij} f_{ij} = \sum_{\forall k | \exists jk} f_{jk}$, where the left hand side of the relation sums the flows entering node *j* and the right hand side sums the flows leaving it.

Given these constraints, and the obvious one that flows are non-negative (i.e $f_{ij} \geq 0$ ), we can write the objective function as trying to maximize the flow leaving $s$. That is $O = \sum_{\forall i | \exists si} f_{si}$. The LP is now trivial to frame. It consists of the capacity inequalities (written as equalities after introducing slack variables), the conservation conditions (with the right hand side carried over to the left and adding slack variables), and the conditions on the flows being non-negative. Some examples of these constraints are:

On edge 5-7, $f_{57} \leq 5.8$ and at vertex 3, $f_{23} + f_{53} = f_{34}$ .

## 4.6 Integer linear programming

Linear programming allows variables to assume real values. In Integer Linear Programming, or ILP, variables are only allowed to assume integer values. Although this may appear to be a small difference, this difference makes the solution of ILP *much* harder. More precisely, though LP can be solved in time polynomial to the size of the input (i.e., it is in P), no polynomial-time solution to ILP is known: on some inputs, an ILP solver can take time exponential in the size of the input (i.e., it NP-hard). In practice, this means that LP can be used to solve problems with hundreds of thousands of variables, but solving an ILP may take a long time even with a few tens of variables.

Nevertheless, ILP arises naturally in a number of cases. In networking, the most common use of ILP is for the *scheduling* problem, where we need to assign discrete time slots to job requests. Since requests cannot be allocated fractional time slots, the problem is naturally posed as one with integer constraints, as the next example shows.

**Example 6 (A scheduling problem)**

Two users, Alice and Bob, can schedule jobs on a machine in one of two time periods, Period 1 or Period 2. If Alice schedules a job during Period 1, she gains a benefit of 20 units, and incurs a cost of 10 units, and during Period 2, she gains a benefit of 10 units and incurs a cost of 20 units. If Bob schedules a job during Period 1, he gains a benefit of 100 units and incurs a cost of 10 units, and during Period 2, he gains a benefit of 10 units and incurs a cost of 200 units. Each user may schedule at most one job in one time unit and in each time period, at most one job can be scheduled.

Express this system in standard form to maximize the benefit derived from the assignment of user jobs to time periods (also called a *schedule*).

The control parameters here are the choice of assignments of user jobs to time periods.We have four jobs and only two time periods. Let $x_{ij}$ be a control parameter that is set to 1 if user $i$ is assigned to schedule a job in time period $j$. A user can schedule at most two jobs (one in Period one, and one in Period two), so we have:

$$x_{11} + x_{12} \leq 2$$

$$x_{21} + x_{22} \leq 2$$

In each time period, we can have at most one job scheduled, so we have:

$$x_{11} + x_{21} \leq 1$$

$$x_{12} + x_{22} \leq 1$$

If Alice's job is scheduled in Period 1, the net benefit is (20-10) and if it isn't, the benefit is 0.

We can express the benefit to Alice in time Period 1, therefore, as:

$x_{11}(20\text{-}10)$

Similarly, taking into account the other costs and benefits, the overall objective function is:

$$O = x_{11}(20-10) + x_{12}(10-20) + x_{21}\ (100-10) + x_{22}(10-200)$$

$$= 10x_{11} -10\, x_{12} + 90x_{21} -190x_{22}$$

We want to maximize *O*, but in the standard form, we seek to minimize the objective function. Thus, we simply define the new objective function $O^* = -O$.

Note that the constraints are not in standard form because of the inequalities. We can rewrite the constraints using the slack variables $s_1$ - $s_4$ as:

$$x_{11}+x_{12}+s_1 = 2$$

$$x_{12}+x_{22}+s_2 = 2$$

$$x_{11}+x_{21}+s_3 = 1$$

$$x_{12}+x_{22}+s_4 = 1$$

or

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{11} \\ x_{12} \\ x_{21} \\ x_{22} \\ s_1 \\ s_2 \\ s_3 \\ s_4 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 1 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x_{11} \\ x_{12} \\ x_{21} \\ x_{22} \\ s_1 \\ s_2 \\ s_3 \\ s_4 \end{bmatrix} \ge \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

This expresses the system in standard matrix form. Note that all the $x_{ij}$s are 0 or 1, and therefore the system is an ILP.

[]

When confronted with an ILP, there are three alternatives available to the practitioner. First, to model the system so that the input size is small. This way, an ILP solver will likely not take too long. Second, to look for heuristic solutions that will not find the optimal solution, but are good enough in practice (these approaches are discussed later in this chapter). Finally, if the problem structure allows it, an ILP can be solved by LP. We consider this third alternative next.

### 4.6.1 Total unimodularity

In some cases, we can solve an ILP simply by ignoring the integer constraint and solving it using LP. The solutions will 'magically' be integers! Therefore, its always worth checking if an ILP satisfies this so-called 'total unimodularity' condition.

A square, integer matrix *A* is called *unimodular* if its determinant is either 0, +1 or -1. An integer matrix (which may itself not be square) is called *totally unimodular* if every square, nonsingular submatrix is unimodular.

In practice, there is a simpler test for unimodularity (Theorem 13.3 from [PS82]):

1. Every entry is either 0, 1, or -1.
2. There are zero, one, or two non-zero entries in any *column*.
3. The *rows* can be partitioned into two sets *A* and *B* such that:

   (a) If a column has two entries of the same sign, one of these is in A, and the other is in B.

   (b) If a column has two entries of different signs, both entries are in either A or B.

**Example 7 (A totally unimodular matrix)**

Here is an example of a totally unimodular matrix:

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 1 \\ 0 & -1 & -1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & -1 & 0 \end{bmatrix}$$

The matrix can be divided into two sets of two rows (the first and second and the third and fourth) that meet the test for unimodularity.

[]

In the matrix formulation of the problem:

$$\boldsymbol{Ax} = \boldsymbol{b}$$

if **A** is totally unimodular, the corresponding ILP can be solved using LP, and the solutions will turn out to be integers.

## *4.7 Using integer linear programming*

Like LP, ILP can also be used to model a variety of problems. As an example, we will study the use of ILP to solve the weighted bipartite matching problem.

### 4.7.1 Weighted bipartite matching

A bipartite graph is a graph where the vertices can be divided into two sets such that all the edges in the graph have one vertex in one set and another vertex in another set. Such graphs arise naturally in many problems. For instance, one set of vertices could represent a set of demands and the other set could represent a set of resources. Edges would then show the resources that can meet each demand. A weight on such an edge could represent the goodness of fit or perhaps the cost of the resource.

**Example 8: (A weighted bipartite graph)**

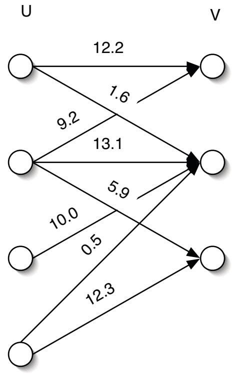


FIGURE 5. **Example of a weighted bipartite graph**

A matching *M* on a graph is the subset of edges such that no two edges in this set share a common vertex. 'Matching' indicates that each edge in *M* matches one vertex in one set (say, a demand) to a vertex in another set (say, a resource). A *maximal weighted* matching is a matching such that the sum of the edge weights, summed over the edges in *M*, is the greatest. If only one such matching exists, it is also the *maximum* weighted matching.

We can use ILP to find the maximal weighted matching in a bipartite graph. Let *U* and *V* be the two sets of vertices and let *uv* refer to an edge that goes from some element *u* in *U* to some element *v* in *V*. Let $w_{uv}$ be the weight of such an edge.

We define integer variables $x_{uv}$ that are set to 1 if the corresponding edge is in the matching *M* and 0 otherwise. Clearly, the total weight of a matching is $\sum_{u,v} w_{uv}x_{uv}$, and this is the value we wish to maximize in the LP (i.e., this is the objective function). Now we need to set up the constraints the right way. We want to ensure that there is at most one edge from a particular element in *U* to any element in *V* and *vice versa* (there may be no edge incident at some node in *U* or *V* in *M* if |*U*| and |*V*| are not equal). It is convenient to convert the original graph, where not every element in *U* has an edge to an element in *V*, to a *complete* graph, where we add extra zero-weight edges so that every element in *U* has |*V*| edges. Then, the constraints are:

$$\forall u \sum_{w} x_{uw} \le 1$$

$$\forall v \sum_{w} x_{wv} \le 1$$

The first constraint ensures that at most one edge in *M* leaves every node in *U*. If this is an extra edge, it adds zero weight to *M*'s weight and can be ignored in the final solution. Similarly, the second constraint ensures that at most one edge is incident at every node in *V*.

## *4.8 Dynamic programming*

We now turn our attention to *dynamic programming,* another powerful technique for optimization. It can be applied to many problems that can be decomposed into sub-problems, such that the optimal solution to the original problem is a composition of the optimal solution to each sub-problem (this is also called the *optimal substructure* of the problem). The technique, then, is to decompose the problem into two or more sub-problems, solve each sub-problem recursively, and then put the solutions together again to obtain the final answer. Of course, the recursion needs to end in a 'small' problem that can be easily solved. Moreover, it is critical that there aren't too many sub-problems.

**Example 9: (Fibonacci computation)**

Although not an optimization problem, the Fibonacci sequence clearly demonstrates the meaning of substructure, composition, and the need to limit the number of sub-problems. This sequence is defined by F(1) = F(2) = 1, and for *n > 2*, F(n) = F(n-1) + F(n-2). The first few terms of the sequence are 1, 1, 2, 3, 5, 8, 13.

Suppose you want to compute F(k). You can decompose this into two sub-problems, computing F(k-1) and computing F(k-2). Then, these solutions are composed simply by adding them. Finally, note that the computation of F(k-1) *reuses* the solution to F(k-2), because F(k-1) = f(k-2) + f(k-3). So, computing F(k-2) is used twice: once to compute F(k) and once to compute F(k-1). In fact, a little thought shows that we compute each sub-problem (that is F(k-*i*) for *i* in 1...k-1) only *once*! This is what makes dynamic programming efficient. If we could not reuse solutions of sub-problems, we would have a trivial divide-and-conquer algorithm that could potentially require the solution of an exponential number of sub-problems.

[]

Dynamic programming is useful only when we can strictly limit the number of underlying sub-problems. Moreover, it is necessary to remember the solution to the sub-problems so that they are not repeatedly re-solved. This is called *memoization.*

There are two standard approaches to dynamic programming. The first is *bottom-up*, where sub-problems are solved starting from the simplest one. They are then composed to find the required solution. In the case of the Fibonacci sequence, this would correspond to computing F(1), F(2) etc. until getting to F(k). In the *top-down* approach we start with the full problem and decompose it as we did in the example.

**Example 10 (Floyd-Warshall algorithm for all-pairs shortest paths)**

A well-known example of the use of dynamic programming in a networking context is the Floyd-Warshall algorithm for simultaneously computing the shortest paths between all pairs of nodes in a graph in only $O(N^3)$ time. This uses the bottom-up approach to dynamic programming.

The algorithm operates in the context of an undirected graph *G* with *N* nodes whose nodes are numbered 1...*N*. Let's define a *path* as a sequence of nodes such that no node index is repeated. The length of the path is the number of edges in it. We want to find the shortest path from any node *i* to any node *j*.

We will consider all paths from node *i* to node *j* that obeys the following non-trivial rule: the path only contains nodes numbered from 1...*k*. Let *s(i,j,k)* denote the shortest of these paths (if no such path even exists, *s* returns infinity). For the moment, we will abuse notation and use *s* to denote both the path and its length.

Now, taking the bottom up approach, assume that you have already computed *s(i,j,k-1)*, that is, the shortest path from *i* to *j* that only uses nodes numbered 1...*k-1*. We'll see how to use this to compute *s(i,j,k)*.

The solution follows from the following observation: either the shortest path from *i* to *j* includes the node numbered *k* or it doesn't. If it doesn't, then *s(i,j,k) = s(i,j,k-1)*. If it does, then there has to be a path from *i* to *j* passing through *k*, which means that *k* must be reachable from both *i* and *j* using only nodes numbered 1...*k-1*. Moreover, the shortest path is composed from the shortest path from *i* to *k* using only nodes 1...*k-1* and the shortest path from *k* to *j* using only nodes 1...*k-1*. We see that *s(i,j,k) = s(i,k, k-1) + s(k, j, k-1)*.

We now have the decomposition we need:

$$s(i,j,k) = min(s(i,j,k-1), s(i,j,k-1)+s(k, j,k-1))$$

We can immediately see that we need to compute the values (and paths) *s(i,j,1)* for all *i,j* bottom up. Once we have these, we can use these to compute *s(i,j,2)* for all values of *i,j* and repeat for increasing values of *k*. Dynamic programming is successful here because of the optimal substructure, the ease of composition, and the limited number of sub-problems.

[]

Although this is not an optimization problem, per se, it demonstrates the meaning of substructure, composition, and the need to limit the number of sub-problems.

## 4.9 Nonlinear constrained optimization

So far, we have been examining the use of optimization techniques where the objective function and the set of constraints are both linear functions. We now consider situations where these functions are not linear.

How does non-linearity change the optimization problem? In a (non-integer constrained) linear system, the objective function attains its maximum or minimum value at one of the vertices of a polytope defined by the constraint planes. Intuitively, because the objective function is linear, we can always 'walk along' one of the hyper-edges of the polytope to increase the value of the objective function, so that the extremal value of the objective function is guaranteed to be at a polytope vertex.

In contrast, with non-linear optimization, the objective function may both increase and decrease as we walk along what would correspond to a hyper-edge (a contour line, as we will see shortly). Therefore, we cannot exploit polytope vertices to carry out optimization. This opens the door to a large number of non-linear optimization techniques, which we study next.

Non-linear optimization techniques fall into roughly into two categories.

- When the objective function and the constraints are mathematically 'nice', that is, continuous and at least twice differentiable, we can use two well-known techniques, Langrangian optimization and Lagrangian optimization with the Karush-Kuhn-Tucker conditions.
- When the objective functions are not continuous or differentiable, we can bring to bear several heuristic techniques such as hill-climbing, simulated annealing, and ant algorithms.

We will first look at Lagrangian techniques (4.10 on page 104), a variant called the KKT conditions that allows inequality constraints (4.11 on page 105) then briefly consider several heuristic optimization techniques (4.12 on page 106).

## 4.10 Lagrangian techniques

Lagrangian optimization allows us to compute the maximum (or minimum) of a function $f$ of several variables subject to one or more constraint functions denoted $g_i$. We will assume that $f$ and all the $g_i$ are continuous, at least twice-differentiable, and are defined over the entire domain, that is, do not have boundaries.

Formally, $f$ is defined over a vector **x** drawn from $R^n$ and we want to find the value(s) of **x** for which $f$ attains its maximum or minimum, subject to the constraint function(s): $g_i(\mathbf{x}) = c_i$, where the $c_i$ are real constants.

To begin with, consider a function $f$ of two variables $x$ and $y$ with a single constraint function. We want to find the set of tuples of the form $(x,y)$ that maximize $f(x,y)$ subject to the constraint $g(x,y) = c$. The constraint $g_i(\mathbf{x}) = c_i$ corresponds to a *contour* or *level set*, that is, a set of points where $g$'s value does not change. Imagine walking along such a contour. As we do so, $f$ will increase and decrease in some manner. Imagine the contours of $f$ corresponding to $f(\mathbf{x}) = d$ for some value of $d$. We can think of the walk on $g$'s contour touching successive contours of $f$. An extremal value of $f$ on $g$'s contour is reached exactly when $g$'s contour grazes an extremal contour of $f$. At this point, the two contours are tangential, so that the gradient of $f$'s contour (a vector that points in a direction perpendicular to the contour) has the same direction as the gradient of $g$'s contour (though it may have a different absolute value). More precisely, if we denote the gradient by $\nabla_{x,y} = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right)$, then at the constrained extremal point, $\nabla_{x,y}f = \lambda\nabla_{x,y}g$.

We define an auxiliary function:

$$F(x, y, \lambda) = f(x, y) + \lambda(g(x, y) - c)$$

A little thought shows that the stationary points of *F*, that is the points where $\nabla_{x, y, \lambda}F(x, y, \lambda) = 0$, are points that satisfy the constraint *g*, because the partial derivative with respect to $\lambda$ has to be zero, and are also extremal points, because partial derivatives in the *x* and *y* directions are 0. Thus, the extremal points of *F* are the points of constrained extrema (i.e minima or maxima).

From Fermat's theorem, the maximum or minimum value of any function is attained at one of three types of points: (a) a boundary point (b) a point where *f* is not differentiable and (c) at a stationary point where its first derivative is zero. Because we assume away the first two situations, we know that the maximum or minimum is attained at one of the stationary points of *F*. Thus, we can simply solve for $\nabla_{x, y, \lambda}F(x, y, \lambda) = 0$ and use the second derivative to determine the type of extremum.

This analysis continues to hold for more than two dimensions and more than one constraint function. Thus, from a 'plug-and-chug' point of view, we simply take the objective function and add to it a constant times each constraint function to get the auxiliary. Then, we solve the system of equations obtained by setting the gradient of the auxiliary function to 0 to find the stationary points (i.e. the constrained extrema).

**Example 11: (Lagrangian optimization)**

Consider a company that purchases capacity on a link to the Internet and has to pay for this capacity. Suppose that the cost of a link of capacity *b* is *Kb*. Also suppose that the mean delay experienced by data sent on the link, denoted by *d,* is inversely proportional to *b,* so that *bd = 1*. Finally, let the benefit *U* from using a network connection with capacity *b,* and delay *d* $0 \le d < \infty$ be described by *U = -Kb -d*, that is, it decreases both with cost and with the delay. We want to maximize *U* subject to the constraint *bd = 1*. Both *U* and the constraint function are continuous and twice-differentiable. Therefore, we can define the auxiliary function:

$$F = -Kb - d + \lambda(bd - 1)$$

We set the partial derivatives with respect to *b, d* and $\lambda$ to zero, to obtain, respectively:

$$\begin{aligned} -K + \lambda d &= 0 \\ -1 + \lambda b &= 0 \\ bd &= 1 \end{aligned}$$

From the second equation, we see that $b = 1/\lambda$ and from the first equation, $d = K/\lambda$ and from the third equation, substituting the values for *b* and *d*, $\lambda = \sqrt{K}$ . Putting this back into the equations for *b* and *d,* we find $b = 1/(\sqrt{K})$ and $d = \sqrt{K}$ .This gives a value of *U* at (*b*, *d*) to be $-2\sqrt{K}$ . Since *U* is clearly unbounded in terms of a smallest value (when *b* approaches 0), this is also its maximum.

[]

## *4.11 Karush-Kuhn-Tucker conditions for nonlinear optimization*

The Lagrangian method is applicable when the constraint function is of the form *g(**x**) = 0*. What if the constraints are of the form $g(x) \le 0$ ? In this case, we can use the Karush-Kuhn-Tucker conditions, often called the KKT or Kuhn-Tucker conditions, to determine whether the stationary point of the auxiliary function is also a global minimum.

As a preliminary, we need to define what is meant by a *convex* function. A function *f* is convex if, for any two points *x* and *y* in its domain, and for *t* in the closed interval [0,1], $f(tx + (1-t)y) < tf(x)) + (1-t)y$. That is, the function always lies below a line drawn from *x* to *y*.

We consider a convex objective function $f\colon R^n \to R$ with both *m* inequality and *l* equality constraints. We denote the inequality constraints by $g_i(\mathbf{x}) \le 0,\ 1 \le i \le m$ and the equality constraints by $h_j(\mathbf{x}) = 0,\ 1 \le j \le l$. The KKT conditions require all the $g_i$ to be convex and all the $h_j$ to be linear. Then, if *a* is a point in $R^n$, and there exist *m* and *l* constants respectively, denoted $\mu_i$ and $\nu_j$ such that the following conditions hold, then we can guarantee that *a* is a globally constrained minimum of *f*:

$$\nabla f(a) + \sum_{i=1}^{m} \mu_i \nabla g_i(a) + \sum_{j=1}^{l} \mu_j \nabla h_j(a) = 0$$

$$g_i(a) \le 0 \forall i$$

$$h_j(a) = 0 \forall j$$

$$\mu_i \ge 0 \forall i$$

$$\mu_i g_i(a) = 0 \forall i$$

If these conditions are met, then the stationary points of the auxiliary function (which is the first equation above) yield the minima of *f*.

## 4.12 Heuristic optimization

We now turn to the situation where the objective function and the constraints may not be mathematically ‘nice,’ that is, linear or convex. In such cases, we need to rely on heuristic approaches to optimization. Many heuristic approaches have been proposed in the literature: we will outline only two common ones- Hill climbing and Genetic algorithms.

### Hill climbing

Hill climbing is perhaps the simplest technique for heuristic optimization. In its simplest variant, it does not even support constraints, seeking only to find the value of **x** that minimizes *f(**x**)*. Hill climbing requires only two primitives: a way to evaluate *f(**x**)* given **x** and a way to generate, for each point **x,** another point **y,** that is ‘near’ **x** (we assume that **x** and **y** are embedded in a suitable metric space).

We start by randomly choosing a point **x** in the domain of *f* and labelling it the candidate maximum (we might just get lucky!). We evaluate *f* on **x**, then generate a point **y** that is ‘close’ to **x**. If the value of *f* is higher at **y,** then **y** is the new candidate maximum, else **x** remains the candidate. We continue to generate and evaluate *f* on neighbouring points of the candidate maximum until we find a point **x** all of whose neighbours have a lower value of *f* than at **x**. We declare this the maximum.

The analogy to climbing a hill is clear. We start somewhere on the hill and take a step in a random direction. If it’s higher up, we step up. If not, we stay where we are. This way, assuming the hill has a single peak, we will eventually get to the top, where every neighbouring step takes us downhill.

Although simple, this approach to hill climbing leaves much to be desired. These concerns are addressed by variants of the basic approach:

- We could generate more than one neighbour of **x** and choose the best of these. This variant is also called the *steepest-gradient method*.
- We could memorize some or all of the values of **y** that we did not end up taking in a *tabu list*. Then, if we generated any one of these values again, we could immediately discard it. This variant is called *tabu search*.

- If we wanted to find the maximum value of *f* subject to constraint *g*, we could simply choose the initial candidate maximum **x** to be a value that also satisfied *g*. Then, when generating neighbours of **x**, we ensure that the neighbouring values also satisfied *g*. This allows us to use hill climbing for constrained optimization.

The single biggest problem with hill climbing is that it fails badly when *f* has more than one maximum. In this case, an unfortunate initial choice of **x** will cause the algorithm to get stuck in a local maximum, instead of finding the global maximum. This is illustrated in Figure 6 which shows a function with multiple peaks. Starting at the base of any of the lesser peaks will result in hill climbing stopping at a local maximum.

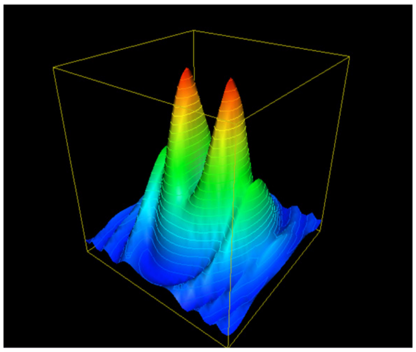


FIGURE 6. **Example of a function where hill climbing can get stuck in a local maximum**

There are several ways to get around this problem. One approach is called *shotgun* hill climbing. Here, the hill climbing algorithm is started from several randomly chosen candidate maxima. The best result from among these should be the global maximum as well. This approach is widely used.

A second approach, called simulated annealing, varies the 'closeness' of a selected neighbour dynamically. Moreover, it allows for some steps of the climb to be downhill. The idea is that if the algorithm is stuck at a local maximum, the only way out is to go down before going up, therefore downhill steps should be allowed. The degree to which downhill steps are permitted varies over the climb. At the start even large downhill steps are permitted. As the climb progresses, however, only small downhill steps are permitted.

More precisely, the algorithm evaluates the function value at the current candidate point **x** and at some neighbour **y**. There is also a control variable, called the temperature, *T*, that describes how large a downhill step is permitted. The *acceptance function A(f(**x**), f(**y**), T)* determines the probability with which the algorithm moves from **x** to **y** as a function of their values and the current temperature, with a non-zero probability even when *f(**y**) < f(**x**)*. Moreover, the acceptance function tends to zero when T tends to zero and *f(**y**) < f(**x**)*. The choice of the acceptance function is problem-specific and therefore usually handcrafted.

## Genetic algorithms

The term 'genetic algorithm' applies to a large class of approaches that share some common attributes. The key idea is to encode a candidate maximum value **x** as a bit string. At the start of the algorithm, hundreds or thousands of such values are randomly generated. The function *f* is then evaluated at each such value, and the best ones are selected for propagation. Propagation occurs in two ways. With *mutation*, some bits of a candidate value are randomly perturbed to form the next generation of candidates. With *crossover*, bits from two candidate values are randomly exchanged. In this way, the best 'features' of the population are hopefully carried over to the next generation. The algorithm proceeds by forming generation after generation of candidates, until adequate solutions are found.

This approach has many tuning parameters and there is an extensive literature on how to encode candidates, how to introduce mutations, and how to make effective crossovers. In the networking literature, genetic algorithms are known for some scheduling problems.

## 4.13 Exercises

**1 Modelling**

You have been hired as the head of CHYM FM’s balloon operations. Too much money is being spent for each flight! Your job is to make flight profitable again (the number of flights is not negotiable).

For each flight, you can control where you take off from (there is a finite set of take-off locations) and the duration of the flight, as long as the flight lasts at least 15 minutes. The cost of a flight depends on its duration (to pay for natural gas, the pilot’s wages, and for the chase vehicle), where the balloon takes off from, and how far the landing site is from a road (the further away it is from a road, the more it has to be dragged over a farmer’s field). Moreover, you can have up to 9 passengers (in addition to at least one pilot), and charge them what you wish. Of course, the number of passengers decreases (say linearly) with the cost of the ticket.

What are the fixed parameters? What are the input and output parameters? What are the control variables? Come up with plausible transfer and objective functions. How would you empirically estimate the transfer function?

**2 Optimizing a function of two variables**

Consider the following system:

$$O = 10x_1 - 3x_{2,} \text{ where}$$

$$2x_1 - x_2 = 1 \quad \textit{and}$$

$$x_1 \geq 0 \; x_2 \geq 0$$

Geometrically find the optimal value of *O*.

**3 Optimizing a function of three variables**

Geometrically find the optimal value of *O* where

$$O = 5x_1 + 2x_{2,} - x_3 \text{ and} \qquad \textbf{(EQ 7)}$$

$$x_1 + x_2 + x_3 = 1 \text{ and} \qquad \textbf{(EQ 8)}$$

$$x_1 \geq 0 \; x_2 \geq 0 \quad 3 \geq 0 \qquad \textbf{(EQ 9)}$$

**4 Network flow**

Model the network flow problem of Example 5, where the warehouses have finite bounded capacity, as a linear program.

**5 Integer linear programming**

Generalize Example 6 to the case where *n* users can schedule jobs on one of *k* machines, such that each user incurs a specific cost and gains a specific benefit on each machine at each of *m* time periods. Write out the ILP for this problem.

**6 Weighted bipartite matching**

Suppose you have *K* balls that need to placed in *M* urns such that the payoff from placing the *k*th ball in the *m*th urn is $p_{km}$, and no more than 2 balls can be placed in each urn. Model this as a weighted bipartite matching problem to maximize the payoff.

**7 Dynamic programming**

You are given a long string *L* of symbols from a finite alphabet. Your goal is to find the matching substrings of *L* with a shorter string *S* from the same alphabet. However, matches need not be exact: you can delete one element of *L* or *S* for a penalty of 1, and you can also substitute an element of *L* for an element of *S* for a penalty of 1. So, for example, the match between the string *L* = “text” and *S* = “tx” is “te” with a penalty of 1 (one substitution), “tex” with a penalty of 1 (one deletion), “ex” with a penalty of 2 (two substitutions), “ext” with a penalty of 2 (one substitution and one deletion) etc. Use dynamic programming to output all matching substrings of *L* along with the corresponding penalty.

### 8 Lagrangian optimization

Use Lagrangian optimization to find the extremal values of $z=x^3 + 2y$ subject to the condition that $x^2+y^2 = 1$ (i.e. the points $(x,y)$ lie on the unit circle.

### 9 Hill climbing

Suppose you know that the objective function you are trying to maximize has no more than *K* local optima. Outline an algorithm that is guaranteed to find the global optimum using hill climbing.

CHAPTER 6

# Foundations of Queueing Theory

## 6.1 Overview

Queues arise naturally when entities demanding service interact asynchronously with entities providing service. Service requests may arrive at a server when it is either unavailable or busy serving other requests. In such cases, service demands must be either queued or dropped. It is better to queue demands instead of dropping them when possible. This allows us to smooth over fluctuations both in the rate of service and in the rate of request arrivals. This leads to the creation of a queueing system. The study of the probabilistic behavior of such systems is the subject of queueing theory.

**Example 1:1: (Examples of queueing systems)**

Here are some examples of queueing systems:

4. The arrival of packets to the output queue of a switch. Packets may arrive when the output link is busy, in which case the packets (implicitly, service requests) must be buffered (queued).
5. The arrival of HTTP requests to a web server. If the web server is busy serving a request, incoming requests are usually queued.
6. The arrival of telephone calls to a switch processor. The processor may be unable to service the call because the switch is busy. In this case, the call is queued, awaiting the release of network resources.

[]

Given a queueing system, we would like to obtain certain quantities of interest, such as:

- The expected waiting time for a service request (the *queueing delay*).
- The mean number of service requests that are awaiting service (the *backlog*).
- The fraction of time that the server is busy (or idle).
- The fraction of service requests that must be dropped because there is no more space left in the queue (the *drop rate*).
- The mean length of a busy (idle) period.

Queueing theory allows us to compute these quantities--both for a single queue and for interconnected networks of queues--as long as the incoming traffic and the servers obey certain strict conditions. Unfortunately, it is well established that traffic in real networks do *not* obey these conditions. Worse, we cannot mathematically analyse most networks that are subjected to realistic traffic workloads. Nevertheless, it is worth studying queueing theory for two important reasons. First, it gives us fundamental insights into the behavior of queueing systems. These insights apply even to systems that are mathematically intractable. Second, the solutions from queueing theory from unrealistic traffic models are a reasonable first approximation to reality. Therefore, as long as we keep in mind that results from queueing theory are only approximate and are primarily meant to give an insight into a real system, we can derive considerable benefit from the mathematical theory of queues, which are the subject of this chapter.

To be consistent with the literature, we will use the standard terminology from Kleinrock's Queueing Systems (Vol. 1).

### 6.1.1 A general queueing system and notation

We now introduce some terminology and notation that can be used to describe all queueing systems.

A queue arises when *customers* present *service requests* or *jobs* to one or more *servers*. Customers arrive at times *t* and the time between arrivals is described by the *interarrival time distribution* $A(t) = P(\text{time between arrivals} \le t)$. Similarly, we denote the service time by *x*, which has a *service time distribution* $B(x) = P(\text{service time} \le x)$. Note that we will almost always consider a single queue in isolation.

### 6.1.2 Little's law

We already have enough terminology to prove a fundamental result that holds true for *all* arrival and service processes. It was first proven by J.D.C. Little in 1951. This law states that the mean number of customers in the system is the product of their mean waiting time and their mean arrival rate. This is either obvious or deep!

**Example 2: (Using Little's law)**

Suppose you receive email at the average rate of one message every five minutes. If you read all your incoming mail once an hour, what is the average time that a message remains unread?

*Solution*

The message arrival rate is 12 messages/hour. Because you read email once an hour, the mean number of unread messages is 6. By Little's law, this is the product of the mean arrival rate and the mean waiting time, which immediately tells us that the mean time for a message to be unread is 6/12 hours = 30 minutes.

[]

**Example 3: (Another application of Little's law)**

Suppose that 10,800 HTTP requests arrive to a web server over the course of the busiest hour of the day. If we want to limit the mean waiting time for service to be under 6 seconds, what should be the largest permissible queue length?

*Solution*

The arrival rate $\lambda$ = 10,800/3600 = 3 requests/second. We want $T \le 6$. Now, $N = \lambda T$, so $T=N/\lambda$. This means that $N/\lambda \le 6$ or that $N \le 6*3 = 18$. So, if the mean queueing delay is to be no larger than 6 seconds, the mean queue length should not exceed 18. To be conservative, the web server could return a server busy response when the queue exceeds 18 requests (why is this conservative?)

[]

**Proof of Little's law**

Suppose customers arrive at a queue at a mean rate of $\lambda$ customers/second. Then, in a time interval of length *t* seconds*,* there will be an average of $\lambda t$ arrivals. Let *T* denote the mean waiting time of a customer in seconds. The total time spent waiting in the queue across all customers during the time interval is therefore $\lambda Tt$ customer-seconds.

Let *N* denote the mean number of customers in the queue. In one second, these *N* customers accumulate *N* customer seconds of total waiting time. Thus, in *t* seconds, they accumulate a total of *Nt* customer-seconds of waiting time. This must equal $\lambda Tt$, which implies that $N=\lambda T$.

Note that this argument applies independent of the length of the time interval *t.* Moreover, it does not depend on the order of service of customers, or the number of servers, or on the way in which customers arrive. Thus, it is a powerful and general law applicable to all queueing systems.

## *6.2 Stochastic processes*

Queueing theory is built on the basis of a mathematical construct called a *stochastic process*, which is used to model the arrival and service processes in a queue. We will both intuitively and mathematically define a stochastic process and then study some standard stochastic processes.

**Example 4: (Deterministic and stochastic processes)**

Consider a staircase with 10 steps numbered 1 through 10 and a person standing on the first step, which is numbered 1. Suppose that there is a clock next to the staircase that ticks once a second starting at time 0. Finally, assume that it takes zero time to climb each step.

If the person were to climb one step at each clock tick, then we can predict exactly where the person would be at each time step. At time 0, the person is on step 1, and he stays there until just before time 1. When the clock ticks and time increments to 1, the person would be at step 2 and he stays there until just before time 2. At time 2 he would be on step 3, and so on. We therefore call the act of climbing the staircase in this fashion a *deterministic* process.

We can make things a little more complex. We will allow the person to either climb up one step or down one step or stay on the same step, with some probability of making each choice (other than at the top and bottom, of course). With this change, we lose predictability. That is, we no longer know exactly where the person will be at any moment in time: we can only attach probabilities to the set of places where the person could be at that time. The process is no longer deterministic: it is *stochastic*.

We capture this underlying randomness by means of a random variable $X_i$, where the subscript refers to the *i*th clock tick, and the value of the random variable is the probability distribution over the steps where the person could be at that time. For instance, at time 0, the person is at step 1 with probability 1, so the distribution of $X_0$ over {1, 2,...,10} is {1.0, 0,..., 0}. At time 1, the person is on step 2 with probability 1, so the distribution of $X_1$ over {1, 2,...,10} is {0, 1.0, 0,..., 0}. Suppose that the probability that the person goes up is *p*, that the person goes down is *q,* and that the person stays on the same step is *1-p-q*, except at step 1, where the probability of going up is *p*, and the probability of staying on the same step is *1-p*. Then, at time 2, the distribution of $X_2$ is {*q*, *1-p-q, p*, 0,...,0}. Continuing, at time 3, the distribution of $X_3$ is {*q(1-p), qp + (1-p-q)(1-p-q)+pq, (1-p-q)p+p(1-p-q), pp,* 0,...,0} (Why?)

Generally speaking, at time *i*, we examine the distribution of $X_{i-1}$ and compute the different ways we can reach each step, summing across the probability that each way is taken. This allows us to compute the discrete probability distribution of $X_i$ over the steps.

Note that we must distinguish between the distribution of the random variables at each time step and the actual trajectory taken by a person. For a given trajectory, at each time instant, the person is, of course, only on one step of the staircase. The trajectories are, however, created by sampling from distributions corresponding to each $X_i$. A trajectory is also therefore called a *sample path*. To reiterate, at time step *i*, we know the actual trajectory of the process until that time step and we have probability distributions for the random variables that correspond to the remainder of the trajectory.

[]

This example motivates the following definition of a *stochastic process*: it is a family of random variables $X_i$ that are indexed by the time index $i$. The value of the random variable (in a particular trajectory) is also called the *state* of the stochastic process at that point in time. Without loss of generality, we can think of the states as being chosen from the integers from 1 to $N$. Thus, we can imagine the process as 'moving' from the state corresponding to the value taken by random variable $X_i$ in a given trajectory to the state corresponding to the value taken by random variable $X_{i+1}$ at time $i+1$, just like the person moves from one stair to another. As we have shown, given the probabilities of moving from one step to another, we can, in principle, compute the distribution of each $X_i$: this is the distribution of the stochastic process over the state space at that time.

Time is discrete in this example. In general, time can be continuous. In this case, the family of random variables corresponding to the stochastic process consists of the variables $X(t_1), X(t_2),...$ which represent the (distribution over the) states of the stochastic process at times $t_1, t_2,...$. Given the probability of moving from step to step, we can compute the distribution of $X(t_{i+1})$ from the distribution of $X(t_i)$.

In the example, the person's movements were limited to moving up or down one step on each clock tick. We could, instead, allow the person to go from a given step to any other step (not just the steps above and below) with some probability. Indeed, this distribution of probabilities could differ at different steps and even differ at each clock tick! And, finally, the person could be on a ramp, so that the amount of movement could be a real positive or negative quantity, rather than an integer. These variations are all within the scope of definition of a stochastic process, but the analysis of the corresponding processes is progressively more difficult. We will first describe some possible types of stochastic processes and then focus on the simplest ones.

### 6.2.1 Discrete and continuous stochastic processes

A stochastic process can be discrete or continuous in two dimensions: the values assumed by the random variables (also called the state space) can be discrete or continuous, and the index variable, i.e., time, can also be discrete or continuous.

A *discrete-space process* is one where the random variables $X_i$ take on discrete values.

**Example 5: (Discrete-state process)**

Continuing with Example 4, we see that the set of possible states is the set of stairs, which forms a discrete set.

[]

Without loss of generality, we can think of the state in a discrete-state process as being indexed by an integer in the range 1,2,...$N$.

A *continuous-space process* is one where the random variables take on values from a finite or infinite continuous interval (or a set of such intervals).

**Example 6: (Continuous-state process)**

Continuing with Example 4, consider a person walking up and down a ramp, rather than a stair. This would allow movements by real amounts. Therefore, the random variable corresponding to the state of the process can take on real values, and the corresponding stochastic process would be a continuous-space process.

[]

In a *discrete-time* process, the indices of the random variables are integers. We can think of the stochastic process (in a particular trajectory) as moving from one state to another at these points in time.

**Example 7: (Discrete-time process)**

Continuing with Example 4, this corresponds to a person moving from one step to another exactly at each clock tick. Such a stochastic process is also called a *stochastic sequence*.

[]

In a *continuous-time* process, the times when the process can move to a new state are chosen from a real interval

**Example 8: (Continuous-time process)**

Continuing with Example 4, this corresponds to a person moving at will, independent of the clock ticks.

[]

All four combinations of {discrete space, continuous space} {discrete time, continuous time} are allowed.

### 6.2.2 Markov processes

An important aspect of a stochastic process is how the probability of transitioning from one state to another is influenced by past history. Continuing with our staircase example (a discrete time and discrete space stochastic process), consider a person who is allowed to go from any stair to any other stair. Moreover, we will ask them to obey the following rules: if they arrive at stair 5 from stair 6, then they move to stair 3 next. If, however, they arrive at stair 5 from stair 3, they move to stair 9 next. In all other cases, they move to stair 1 next. Suppose at some point in time we see that they are on stair 5. Where will they go next?

The answer is: we don't know. It depends on where there were in the previous time step. Stated more precisely, the distribution of the random variable *X(n+1)* when *X(n) = 5* depends also on the value of *X(n-1)*. Generalizing from this example, we can come up with rules where the distribution of *X(n+1)* depends not only on *X(n)*, but also on *X(n-1), X(n-2),..., X(1)*. Such systems are inherently complex and there is little we can say about them.

In an attempt to curb this complexity, consider the following rule:

> The distribution of X(n+1) depends only on the value of X(n)

This rule simplifies the situation: if the person is on step 5 at time *n*, then we know the distribution of *X(n+1) no matter how the person got to stair 5*. As we will see, this allows us to compute many quantities of interest about the process. Moreover, many naturally occurring stochastic processes obey this rule. Due to these two facts, stochastic processes that obey this rule are given a special name: they are called *Markov* processes, in honour of A.N. Markov, who first studied them in 1907.

Formally, we state the property as (for the case of discrete time stochastic processes):

$$P(X_{n+1} = j | X_n = i_n, X_{n-1} = i_{n-1}, X_{n-2} = i_{n-2}, X_{n-3} = i_{n-3}, \ldots, X_1 = i_1) = P(X_{n+1} = j | X_n = i_n) \quad \textbf{(EQ 10)}$$

The conditional probability $P(X_{n+1} = j | X_n = i_n)$ is called the *transition probability* to go from state $i_n$ to state *j*.

**Example 9: (Markov process)**

Consider a discrete-time discrete-space stochastic process whose state space is {1,2,3}. Consider the situation at time 2. Let $P(X_3=1 \mid X_2=1) = 0.2$; $P(X_3=2 \mid X_2=1) = 0.4$; $P(X_3=3 \mid X_2=1) = 0.4$. So, if we know that at time 2 $X_2$ is 1, we know the distribution for $X_3$. Moreover, this distribution is independent of prior values of *X*. Assuming that the distributions for $X_3$ when $X_2$ is 2 or 3 are similarly independent of past history, this stochastic process is a Markov process.

[]

Note that at time step *n*, we already know the past history, that is the sequence of prior states $i_1$-$i_n$. So, at time *n*, we are only interested in computing the distribution of the next random variable in the sequence, given this past history. The Markov property allows us to forget everything about history except the value of the current random variable ($i_n$), which encapsulates all past history. This is similar in spirit to the memorylessness property of the exponential distribution that we saw in Section A2.3.2.

A similar property holds true for continuous time stochastic processes. For simplicity, we will first study discrete-time processes that obey the Markov property, also known as *discrete time Markov chains*, before considering continuous time Markov processes.

### 6.2.3 Homogeneity, state transition diagrams, and the Chapman-Kolmogorov equations

For a Markov process, given the value that $X_{n-1}$ attains, we know the probability that $X_n$ takes on each possible value $j$ (these are the transition probabilities associated with the process). This is not as simple as it sounds. Consider the following situation: suppose that, continuing with the example, when the person is on step 4 at time 10, with probability 1 they go to step 5, but when they are on the same step at time 20, with probability 1 they go to step 6. Knowing the value of $X_{10}$ (i.e. 4), we know the distribution of $X_{11}$. Similarly, knowing the value of $X_{20}$ (i.e. 4), we know the distribution of $X_{21}$, so this stochastic process is Markov. However, the probability of going from step 4 to step 5, or going from step 4 to step 6 is time-dependent. Such a process is called a *non-homogeneous Markov process*. For such a chain, the probability that $X_n=j$ is time-dependent. Of course, this complicates the analysis.

Again, we can simplify the analysis by decreeing that the transition probabilities $P(X_n=j|X_{n-1}=i_{n-1})$ should be independent of $n$. In our example, this means that when the person is on a particular step, say step 4, the probability of going to any other step is always the same, no matter when they got to step 4. Such a process is called a *homogeneous Markov process.* For a homogeneous Markov process, we define the quantity time-independent transition probability between state $i$ and state $j$ $p_{ij} = P(X_n=j|X_{n-1}=i_{n-1})$ for any $n$. Note that $p_{ij}$ is independent of $n$.

**Example 10: (Homogeneous Markov process)**

Consider the discrete-time discrete-space stochastic process in Example 9. If this process were to be homogeneous, we need not consider exactly one point in time, such as time 2. Instead, we could let $P(X_{n+1}=1 \mid X_n=1) = 0.2$; $P(X_{n+1}=2 \mid X_n=1) = 0.4$; $P(X_{n+1}=3 \mid X_n=1) = 0.4$. So, if we know that at time $n$ that $X_n$ is 1, we know the distribution for $X_{n+1}$. This generalizes to other possible values for $X_n$.

[]

There are two simple ways to represent state transition probabilities for a homogeneous Markov chain with $N$ states. The first way is in the form of an $N \times N$ transition matrix **P** whose elements are the probabilities $p_{ij.}$ This representation has the attractive property that the probability of going from any state $i$ to state $j$ in two steps is given by the elements of $\mathbf{P}^2$ (Why?). The second way is as a graph (see Example 11). Here, the vertices represent states and the annotation on the edge from vertex $i$ to vertex $j$ is $p_{ij}$. This visually represents a Markov chain. Note that if the chain were non-homogeneous, we would need such a state transition diagram for every time step.

**Example 11: (Representing a homogeneous Markov process)**

Continuing with Example 10: we have already seen that $p_{11} = 0.2$, $p_{12} = 0.4$, $p_{13} = 0.4$. Suppose that $p_{21} = 1.0$, $p_{22} = 0$, $p_{23} = 0$ and $p_{31} = 0.5$, $p_{32} = 0.25$, $p_{33} = 0.25$. Then, we can represent it in two ways as shown below:

$$\boldsymbol{P} = \begin{bmatrix} 0.2 & 0.4 & 0.4 \\ 1.0 & 0 & 0 \\ 0.5 & 0.25 & 0.25 \end{bmatrix}$$

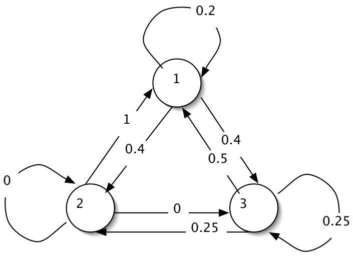


FIGURE 7. **State transition diagram for Example 11**

[]

Given the set of transition probabilities $p_{ij}$ for a homogeneous Markov chain, we can define the *m*-step transition probability from state *i* to state *j* denoted $p^{(m)}{}_{ij}$ by

$$p^{(m)}{}_{ij} = P(X_{n+m}=j|\ X_n=i) = \sum_k p_{ik}{}^{(m-1)} p_{kj} \qquad \text{m=2,3,...} \qquad \textbf{(EQ 11)}$$

where the sum of products form comes from summing across independent events (that of going to some intermediate state in *m*-1 steps), and each term is a product, because it is the combination of two independent events (going from state *i* to state *k* and from state *k* to state *m*). These relations exist only because of the Markovian nature of the chain.They are important enough that they are given their own name: the *Chapman-Kolmogorov* equations. The Chapman-Kolmogorov equations can also be stated as:

$$p^{(m)}{}_{ij} = P(X_{n+m}=j|\ X_n=i) = \sum_k p_{ik} p_{kj}{}^{(m-1)} \qquad \text{m=2,3,... .} \qquad \textbf{(EQ 12)}$$

Comparing the two, we see that in the first formulation we consider going from state *i* to state *k* in *m-1* steps, and from state *k* to state *j* in one step, and in the latter, we go from state *i* to state *k* in one step and from state *k* to state *j* in *m-1* steps.

## 6.2.4 Irreducibility

If every state of a stochastic process can be reached from every other state after a finite number of steps, then the process is called *irreducible*, otherwise it is *reducible*. Moreover, if there are states of a stochastic process can be separated into subsets that are mutually unreachable from each other, we call each such set a *separable sub-chain*.

**Example 12: (A reducible Markov chain)**

Consider the Markov chain in Figure 8. Here, the transition probabilities $p_{ij}$ for *i* even and *j* odd or *i* odd and *j* even are 0. So, if the initial state of the process is an even-numbered state, it will always stay in even-numbered states. Alternatively, if it starts from an odd-numbered state, it will forever stay on odd-numbered states. So, the even-numbered states are unreachable from the odd-numbered steps states and the chain is reducible. Indeed, we could separate out the even-numbered and odd-numbered states into separate chains that would equivalently describe the process. Similarly, we can imagine choosing transition probabilities that allow us to separate a stochastic process into three, four, or more sub-chains.

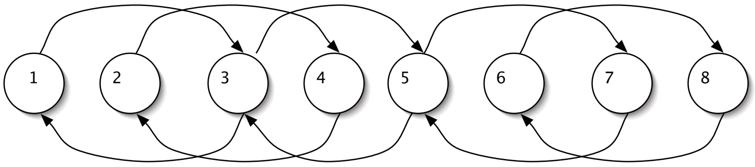


FIGURE 8. **A reducible Markov chain**

[]

### 6.2.5 Recurrence

For every state *j*, one of two conditions hold: after entering state *j*, either the probability of re-entering state *j* after a finite number of steps is 1, or there is some chance that the state is not re-entered after a finite number of steps. In Example 4, this is equivalent to saying that after stepping on a stair, say stair 6, it either certain that the person will return to stair 6, or there is a non-zero probability that the person will not return to stair 6. If return to a state is sure, we call the state *recurrent*, otherwise we call it *transient.*

Let $f_j^n$ denote the probability that the *first* return to state *j* is after *n* steps. State *j* is recurrent if $\sum_{n=1}^{\infty} f_j^n = 1$ and transient otherwise.

Although a state is recurrent, its expected recurrence period may be infinite.The expected recurrence period is defined by $\sum_{n=1}^{\infty} nf_j^n$ . This sum may diverge if $f_j^n$ is sufficiently large for large values of *n*. In such cases, the mean recurrence period is infinite, and the state is called *recurrent null*. Otherwise, it is called *recurrent non-null*.

### 6.2.6 Periodicity

Given a recurrent state *j*, suppose the only way to return to that state is to take *r*, 2*r*, 3*r*... steps, with $r \geq 2$ . We then call the state *j periodic*, with a period *r*. Periodic states arise when the Markov chain has a cycle. A trivial way to check if a state is periodic is to see if it has a self-loop, that is $p_{jj} > 0$. If so, the state can be re-entered with any desired number of steps, which makes *r* =1, and the state *aperiodic*. For an irreducible Markov chain, if *any* state has a self-loop, then all states are aperiodic.

**Example 12: (Periodic and aperiodic Markov chains)**

The Markov chain in Figure 8 is periodic with period 2 (Why?) and the chain in Figure 7 is aperiodic (Why?).

[]

### 6.2.7 Ergodicity

A state *j* is *ergodic* if it is recurrent non-null and aperiodic. Continuing with Example 4, it is a stair that the person will return to (recurrent), with a mean recurrence period that is finite (non-null), and such that the returning times do not have a least common divisor larger than 1 (aperiodic). If all the states in a Markov chain are ergodic, the chain itself is ergodic. It can be shown that a finite aperiodic irreducible Markov chain is always ergodic (in other words, all states of a finite irreducible Markov chain are recurrent non-null).

**Example 13: (Ergodic Markov chain)**

The Markov chain in Figure 7 is finite, aperiodic, and irreducible. Therefore, it is also ergodic.

[]

If a chain is ergodic, then it is insensitive to its initial state. Independent of its initial state, the distribution of $X_n$ (for reasonably large values of *n)* is the same. Even better, suppose we associate some quantity with each state (such as its index). Then, the average quantity measured over some number of steps (the ‘time average’) quickly converges to the average quantity measured over infinitely long sequences of state transitions. This motivates the observation that ergodic chains ‘mix’ well. However, non-ergodic chains are either recurrent null (so that they may take a long time to return to some state), reducible (so some parts of the chain do not communicate with others), or periodic (so that quantities of interest also share the same period). For these reasons, non-ergodic chains do not mix well.

### 6.2.8 A fundamental theorem

We now have enough terminology to state (without proof) a fundamental theorem of queueing theory:

**Theorem 1**: The states of an irreducible Markov chain are either all transient, all recurrent null, or recurrent non-null. If any state is periodic, then all states are periodic with the same period *r*. []

Intuitively, this categorizes all Markov chains into a few types. The first are those where the process goes from state to state but never returns to any state. In this case, all states are transient. In the second and third type of chain the process returns to at least one of the states. But the chain is irreducible, and so we can go from that state to all other states. Therefore, if the process can return to any one state, by definition it can return to all other states, which makes all states recurrent. In the second type, the transition probabilities are such that the expected recurrence period is infinite, so that all states are recurrent null. In the third type, the expected recurrence period for all states is finite. For this type, we have two sub-types: the periodic recurrent non-null chains, whose states all share the same period, and the aperiodic recurrent non-null (ergodic) chains, for whom no such period can be defined.

### 6.2.9 Stationary (equilibrium) probability of a Markov chain

Recall that for a homogeneous Markov chain the state transition probabilities are time-independent. For a homogeneous chain, we expect that the probability of *being* in any particular state to also be time-independent (if the probability of going from one state to another does not depend on time, the probability of being in any state shouldn't either).

Of course, the probability of being in a particular state may depend on the initial state (for non-ergodic chains, that do not forget their initial conditions). Therefore, we define the stationary ('long-term') probability of a Markov chain as follows: Let $\pi_j^{(n)}$ denote the probability that at time step *n* the chain is in state *j* (i.e. P($X_n$=*j*)). Suppose we start with the initial distribution $\pi_j^{(0)} = P_j$. Then, $P_j$ is also the *stationary distribution* if for all *n*, $\pi_j^{(n)} = P_j$. Intuitively, if we start with the probability of being in each state *j* as defined by the stationary distribution, then the transitions from each state according to the transition probabilities do not change the probability of being in each state.

**Example 14: (Stationary distribution)**

Compute the stationary distribution of the Markov chain in Figure 9.

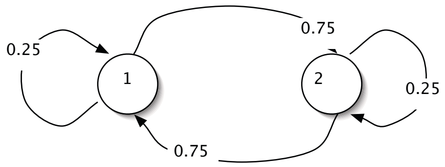


**FIGURE 9.** A simple Markov chain

*Solution*

Suppose that the initial probability of being in state 1 is 0.5 and of being in state 2 is 0.5. After one time step, the probability of being in state 1 is 0.25*0.5 + 0.75*0.5, where the first term is the probability of remaining in state 1, and the second term is the probability of coming from state 2 to state 1, and we sum these probabilities because these are independent events. As expected, this sums to 0.5, so that the probability of being in state 1 in time step 2 is also 0.5. Symmetrically, if the probability of being in state 2 at time 1 is 0.5, the probability of being in state 2 at time 2 is also 0.5. Therefore, the stationary probability distribution of this chain is [0.5 0.5].

[]

### 6.2.10 A second fundamental theorem

We now state a second fundamental theorem that allows us to compute stationary probabilities for any Markov chain.

**Theorem 2**: In an irreducible and aperiodic homogeneous Markov chain, the limiting probabilities:

$$\pi_j = \lim_{n \to \infty} \pi_j^{(n)} \qquad \textbf{(EQ 13)}$$

always exist and are independent of the initial state probability distribution $P_j$. Moreover, if the states are ergodic (being recurrent non-null, in addition to being aperiodic), then $\pi_j > 0$ for all $j$ and can be uniquely determined by solving the following set of equations:

$$\begin{aligned} \sum_j \pi_j &= 1 \\ \pi_j &= \sum_i \pi_i p_{ij} \end{aligned} \qquad \textbf{(EQ 14)}$$

[]

This theorem provides us with a simple set of equations to determine the probability that the Markov chain is in any particular state. We only need verify that the set of states is finite, memoryless (satisfies the Markov property), irreducible (all states can be reached from each other), and aperiodic (for example, because of at least one self-loop). These properties can be verified through simple inspection. Then, we can 'plug and chug' to obtain the probability of being in each state.

**Example 15: (Stationary probability of a Markov chain)**

Compute the stationary probability for the Markov chain in Figure 7.

*Solution*

Note that this chain is ergodic, so we can simply apply Theorem 2 to obtain the following equations:

$$\begin{aligned} \pi_1 &= 0.2\pi_1 + 1\pi_2 + 0.5\pi_3 \\ \pi_2 &= 0.4\pi_1 + 0\pi_2 + 0.25\pi_3 \\ \pi_3 &= 0.4\pi_1 + 0\pi_2 + 0.25\pi_3 \\ 1 &= \pi_1 + \pi_2 + \pi_3 \end{aligned}$$

We solve these to obtain: $\pi_1$=15/31; $\pi_2$=8/31; $\pi_3$=8/31, which is the stationary probability distribution of the chain (Verify this!).

[]

### 6.2.11 Mean residence time in a state

Besides knowing the stationary probability of being in a particular state of a Markov chain, we would also like to know the expected duration that the process spends in each state. This can be computed by first obtaining the probability P(system stays in state $j$ for $m$ additional steps given that it just entered state $j$). The probability that the system stays in the same state after one time step is just $p_{jj}$. Moreover, after one time step, being Markovian, the process has no memory that it was in that state earlier. Therefore, the probability of staying in the state for $m$ steps is given by $p_{jj}^m(1-p_{jj})$, which is a geometrically distributed random variable with parameter ($1-p_{jj}$). This allows us to compute the mean of the distribution, that is, the expected residence time in state $j$, as $1/(1-p_{jj})$.

**Example 16: (Residence time)**

Compute the residence times in each state of the Markov chain shown in Figure 7.

*Solution*

$p_{11}$ = 0.2, so E(residence time in state 1) = 1/0.8 = 1.25.

$p_{22}$ = 0, so E(residence time in state 1) = 1.

$p_{33}$ = 0.25, so E(residence time in state 1) = 1/0.75 = 1.33.

[]

## *6.3 Continuous-time Markov Chains*

Our discussion so far has focused on discrete-time Markov chains, where state transitions happen every clock tick. We now turn our attention to continuous-time chains, where state transitions can happen independent of clock ticks. As we will see, most of the intuitions developed for discrete-time chains carry through to continuous-time chains, with a few minor modifications. The main point of difference is that we need to consider the time instants $t_1$, $t_2$,... when state transitions happen, rather than assuming a state transition occurs at every clock tick. We will briefly state the main results for a continuous-time stochastic process, then focus on a specific type of continuous-time process: the birth-death process.

### 6.3.1 Markov property for continuous-time stochastic processes

We'll first state the Markov property for continuous-time stochastic processes. The stochastic process *X(t)* forms a continuous-time Markov chain if for all integers *n* and for any sequence of times $t_1, t_2,...,t_{n+1}$ such that $t_1 < t_2 <...< t_{n+1}$

$$P(X(t_{n+1})=j \mid X(t_1)= i_1, X(t_2)= i_2,...,X(t_n)= i_n) = \mathrm{P}(X(t_{n+1}=j) \mid X(t_n)= i_n) \qquad \textbf{(EQ 15)}$$

Intuitively, this means that the future ($X(t_{n+1})$) depends on the past only through the current state $i_n$.

The definitions of homogeneity, irreducibility, recurrence, periodicity, and ergodicity introduced for discrete-time Markov chains in Section A3.2 continue to hold for continuous-time chains with essentially no change, so we will not restate them here.

### 6.3.2 Residence time in a continuous-time Markov chain

Analogous to the geometric distribution of residence times in a discrete-time chain, for a continuous-time Markov chain, residence times are exponentially distributed, and for essentially the same reasons. If we denote the residence time in state *j* by $R_j$, the exponential distribution gives us the memorylessness property:

$$P(R_j > s + t | R_j > s) = P(R_j > t) \qquad \textbf{(EQ 16)}$$

### 6.3.3 Stationary probability distribution for a continuous-time Markov chain

We elide the intermediate details and directly present the set of equations necessary to compute the stationary probability of a continuous-time homogeneous Markov chain. We first define the transition probability from state *i* to state *j* by

$$p_{ij}(t) = P(X(s + t) = j | X(s) = i) \qquad \textbf{(EQ 17)}$$

Intuitively, this means that if the process is at state *i* at any time *s,* then the probability that it will get to state *j* after a time interval *t*, is given by $p_{ij}(t)$. This is true for all times because the process is homogeneous.

We also define a new quantity, $q_{ij}$, which denotes the *rate* at which the process departs from state *i* to state *j* (where *j* and *i* differ) when it is in state *i*:

$$q_{ij} = \lim_{\Delta t \to 0} p_{ij}(\Delta t)/\Delta t \qquad \textbf{(EQ 18)}$$

That is, the probability that the process transitions from *i* to *j* during an interval of length *Δt* time units, conditional on it already being at state *i*, is $q_{ij}\Delta t$. Similarly, we define $q_{ii}$ by:

$$q_{ii} = \lim_{\Delta t \to 0} (p_{ii}\Delta t - 1)/\Delta t \qquad \textbf{(EQ 19)}$$

$-q_{ii}$ is the rate at which the process does *not* stay in state *i* (i.e. departs to some other state). Because $\sum_j p_{ij}(t) = 1$ , (at any time *t*, the chain transitions to *some* state, including the current state) we see that

$$\sum_j q_{ij}(t) = 0 \qquad \textbf{(EQ 20)}$$

With these quantities in hand, we can define the time evolution of the probability of being in state *j* at time *t*, defined as $\pi_j(t)$, by:

$$\frac{d\pi_j(t)}{dt} = q_{jj}\pi_j(t) + \sum_{k \neq j} q_{kj}\pi_k(t) \qquad \textbf{(EQ 21)}$$

For ergodic continuous-time Markov chains, as $t \to \infty$ , these probabilities tend to the stationary probability distributions $\pi_j$ which are implicitly defined by:

$$\begin{aligned} q_{jj}\pi_j + \sum_{k \neq j} q_{kj}\pi_k &= 0 \\ \sum_j \pi_j &= 1 \end{aligned} \qquad \textbf{(EQ 22)}$$

(Notice that this is just Equation 21 with the rate of change of the probability set to 0--which is what one would expect for a stationary probability-- and with the time-dependent probabilities replaced by their limiting values).

This ends our brief summary of continuous-time Markov processes. Instead of studying general continuous-time processes, we will instead focus on a smaller, but very important sub-class: that of continuous-time birth-death processes.

## *6.4 Birth-Death processes*

We will restrict our attention in this section to continuous-time homogenous Markov chains that have the property that state transitions are permitted from state *j* only to states *j-1* and *j+1* (if these states exist). This is well-suited to describe processes like the arrival and departure of customers from a queue (the subject of queueing theory, after all!) where the state index corresponds to the number of customers awaiting service. More precisely, if the number of customers in the system is *j*, then the Markov chain is considered to be in state *j*. Customer arrivals cause the number of customers in the system to increase by one, which moves the process to state *j+1* and this happens at a rate $q_{j,j+1}$. Similarly, customer departures (due to service) cause the process to move from state *j* to state *j-1*, and this happens at the rate $q_{j,j-1}$. In keeping with standard terminology, we denote:

$$\begin{aligned} \lambda_j &= q_{j,j+1} \\ \mu_j &= q_{j,j-1} \end{aligned} \qquad \textbf{(EQ 23)}$$

and these are also called the *birth* and *death* rates respectively. Note that these rates can be state-dependent (but cannot be time-dependent due to homogeneity). Also, by definition, the transition rates $q_{ij}$ are 0 for all *j* other than *i, i-1*, and *i+1*. Given this fact, Equation 20, and Equation 23, we find that:

$$q_{jj} = -(\lambda_j + \mu_j) \qquad \text{(EQ 24)}$$

For a birth-death process, being in state *j* has the intuitive meaning that the population size is *j*, that is, there are *j* customers in the queueing system. Note that when *j*=1, we have one customer in the system, the customer that is receiving service, and there are *none* in the queue. Generalizing, in state *j*, we have one customer receiving service and *j*-1 in the queue awaiting service.

### 6.4.1 Time-evolution of a birth-death process

For simplicity, from now on, we will use $P_j(t)$ to refer to $\pi_j(t)$, which is the probability that the population is of size *j* at time *t.* Because a birth-death process is a continuous-time Markov chain, the time evolution of $P_j(t)$ is given by Equation 21. For a birth-death process, we can substitute Equation 23 and Equation 24 to get:

$$\begin{aligned} \frac{dP_j(t)}{dt} &= -(\lambda_j + \mu_j)P_j(t) + \lambda_{j-1}P_{j-1}(t) + \mu_{j+1}P_{j+1}(t) && j \geq 1 \\ \frac{dP_0(t)}{dt} &= -\lambda_0 P_0(t) + \mu_1 P_1(t) && j = 0 \end{aligned} \qquad \text{(EQ 25)}$$

This describes the time-evolution of a birth-death system. In practice, solving these equations is complex, and does not give too many insights into the structure of the system. These are better obtained from the *stationary* probability distribution, which we study next.

### 6.4.2 Stationary probability distribution of a birth-death process

Because a birth-death process is a continuous-time Markov chain, its stationary probability distribution is given by Equation 22. As before, using $P_j$ to refer to $\pi_j$ (the long-term probability of being in state *j*), and substituting Equation 23 and Equation 24 into Equation 22, we obtain the following equations:

$$\begin{aligned} 0 &= -(\lambda_j + \mu_j)P_j + \lambda_{j-1}P_{j-1} + \mu_{j+1}P_{j+1} && j \geq 1 \\ 0 &= -\lambda_0 P_0 + \mu_1 P_1 && j = 0 \\ \sum_j P_j &= 1 \end{aligned} \qquad \text{(EQ 26)}$$

In matrix form, we can write the first two equations as

$$\begin{bmatrix} P_0 & P_1 & P_2 & P_3 & \cdots \end{bmatrix} \begin{bmatrix} -\lambda_0 & \lambda_0 & 0 & \cdots & \cdots \\ \mu_1 & -(\lambda_1 + \mu_1) & \lambda_1 & \cdots & \cdots \\ 0 & \mu_2 & -(\lambda_2 + \mu_2) & \lambda_2 & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix} = 0 \qquad \text{(EQ 27)}$$

where both matrices are infinite-dimensional if the population size is unbounded. Using matrix notation, and defining matrices $\boldsymbol{P}$ and $\boldsymbol{Q}$ appropriately, we can represent this as:

$$\boldsymbol{PQ} = 0 \qquad \text{(EQ 28)}$$

Moreover, defining $\boldsymbol{P(t)}$ appropriately, we can rewrite Equation 25 as

$$\mathrm{d}\boldsymbol{P(t)}/\mathrm{dt} = \boldsymbol{P(t)Q} \qquad \text{(EQ 29)}$$

### 6.4.3 Finding the transition-rate matrix

The ***Q*** matrix defined implicitly by Equation 27 is also called the *transition rate matrix*. It is important because it allows us to derive both the time-dependent evolution of the system (i.e., $P_j(t)$), through Equation 29, and the long-term probability of being in state *j*, through Equation 28. Thus, in practice, the first step in studying a birth-death process is to write down its ***Q*** matrix.

Consider the following representation of a generic birth-death process:

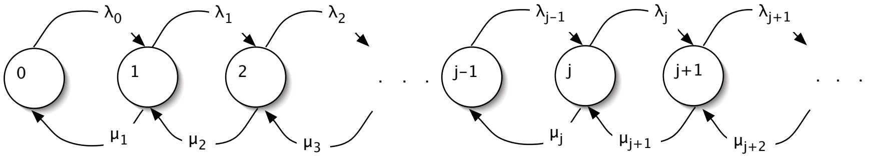


**FIGURE 10. State-transition-rate diagram for a birth-death process**

Here, we represent each state *j* by a circle and we label the arc from state *j* to state *k* with the transition rate $q_{jk}$. From this figure, it's easy to write down ***Q*** for a birth-death process as follows: notice that the diagonal elements of ***Q***, i.e., $q_{jj}$ are the negative of the quantities *leaving* state *j*. Focussing on the *j*th column, the $q_{j-1,j}$th elements, immediately above the diagonal (such as element $q_{01}$), are the rates entering state *j* from state *j-1*, i.e., $\lambda_{j-1}$ and the $q_{j+1,j}$th elements, immediately below the diagonal (such as element $q_{32}$), are the rates entering state *j* from state *j+1*. All other elements are 0. In each row, the quantities sum to zero, due to Equation 20.

Thus, given the state-transition-rate diagram, it is possible to construct ***Q*** and use this to obtain the time-dependent and time-independent (long-term) probabilities of being in each state. We will now use this approach to study some well-known birth-death systems.

We note in passing that this inspection approach applies to all Markov chains, where we can determine the elements of the ***Q*** matrix by inspecting the corresponding state-transition-rate diagram, then solving for ***P*** and ***P***(*t*) using the matrix version of Equation 21 and Equation 22.

**Example 17: (Transition-rate matrix for a birth-death process)**

Consider the state-rate-transition diagram in Figure 11. What are the ***P*** and ***Q*** matrices for this system? What are the equations for its time-evolution and the long-term probability of being in each state?

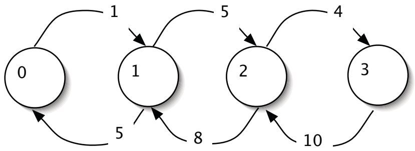


**FIGURE 11. A simple birth-death process**

The ***P*** matrix is $[P_0\ P_1\ P_2\ P_3]$. By inspection, we can write the ***Q*** matrix as

$$\mathbf{Q} = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 5 & -10 & 5 & 0 \\ 0 & 8 & -12 & 4 \\ 0 & 0 & 10 & -10 \end{bmatrix}$$

Thus, the time-evolution of state probabilities is given by:

d$P_0$(t)/dt = -$P_0$(t) + 5$P_1$(t)

d$P_1$(t)/dt = $P_0$(t) - 10$P_1$(t) + 8$P_2$(t)

d$P_2$(t)/dt = 5$P_1$(t) - 12$P_2$(t) + 10$P_3$(t)

d$P_3$(t)/dt = 4$P_2$(t) - 8$P_3$(t)

The long-term probability of being in each state is given by:

-$P_0$ + 5$P_1$ = 0

$P_0$ - 10$P_1$ + 8$P_2$ = 0

5$P_1$ - 12$P_2$ + 10$P$ = 0

4$P_2$ - 8$P_3$ = 0

[]

### 6.4.4 A pure-birth (Poisson) process

Consider a system where $\lambda_j = \lambda$ for all $j$ (the departure rate from all states is the same), and $\mu_j$= 0 for all $j$ (the death rate is 0). This represents a process whose population grows without bound and whose rate of growth is $\lambda$ independent of the population size (that is, we expect the population to grow by 1 every 1/$\lambda$ seconds independent of the current population size). This process is the famous Poisson process. We will consider only two properties of this process here, the probability of being in state $j$ at time $t$, which corresponds to having $j$ arrivals in time $t$, and the distribution of inter-arrival times, that is, the expected time between going from any state to the adjacent state. More sophisticated treatments, found in the references, explore many other properties of this remarkable process.

We can derive the probability of being in any state directly from Equation 25. Substituting the values for $\lambda$ and $\mu$ in this equation, we get:

$$\begin{aligned} \frac{dP_j(t)}{dt} &= -\lambda P_j(t) + \lambda P_{j-1}(t) \qquad j \ge 1 \\ \frac{dP_0(t)}{dt} &= -\lambda P_0(t) \qquad j = 0 \end{aligned} \tag{EQ 30}$$

The second equation is a trivial differential equation whose solution is given by

$$P_0 = e^{-\lambda t} \tag{EQ 31}$$

We substitute this into the first equation to get

$$\frac{dP_1(t)}{dt} = -\lambda P_1(t) + \lambda e^{-\lambda t} \tag{EQ 32}$$

whose solution is

$$P_1 = \lambda e^{-\lambda t} \tag{EQ 33}$$

Substituting this back, as before, and repeating the process, we obtain

$$P_j(t) = \frac{(\lambda t)^j}{j!} e^{-\lambda t} \tag{EQ 34}$$

This, of course, is the density function for the Poisson distribution (see Section 1.4.1) with parameter $\lambda t$. Thus, for a Poisson process with parameter $\lambda$, the probability of $j$ arrivals in time $t$, which is also the probability of being in state $j$ at time $t$, is given by a Poisson distribution with parameter $\lambda t$. Because the mean of the Poisson distribution is also its parameter, we can immediately see that the expected number of arrivals in time $t$ is $\lambda t$. This is intuitively pleasing: the arrival rate is $\lambda$ so in time $t$ we should see, on average, $\lambda t$ arrivals.

**Example 18: (Poisson process)**

Consider students arriving to a class as a Poisson process at a mean rate of 5 students/second. What is the probability that the classroom has 2 and 10 students after 2 seconds?

*Solution*

We have $\lambda = 5$ and $t = 2$, so the Poisson parameter is 10. The probability of having 2 students in the classroom after 2 seconds is $P_2(2) = (10^2/2!)\ e^{-10} = 50*e^{-10} = 2.26*10^{-3}$. This is a very unlikely event.

The probability of having 10 students in the classroom after 2 seconds is $P_{10}(2) = (10^{10}/10!)\ e^{-10} = 0.125$. Note that the expected number of students after 2 seconds is 10, yet the probability that the expected number of students is actually achieved is only one in eight!

[]

We now derive the interarrival time distribution for a Poisson process. Let $a$ denote the continuous random variable that represents the time between any two arrivals: we seek the distribution for $a$. Consider the cumulative density function of $a$, given by the probability P($a \le t$) = 1 - P($a > t$). But P($a > t$) is just P(there are 0 customer arrivals in time (0, $t$)) = 1-$P_0(t)$ = $1 - e^{-\lambda t}$, $t \ge 0$. The density function is given by differentiating this expression to get:

$$a(t) = \lambda e^{-\lambda t} \quad \text{(EQ 35)}$$

We recognize this as an exponential distribution (see Section 1.4.2). This gives us the following important result:

> The interarrival times for a Poisson process are drawn from an exponential distribution.

We note that the exponential distribution is memoryless. Thus, for a Poisson process, not only is the rate of transitioning to the next state (the birth rate) independent of the current population size, the *time* at which this transition occurs does not depend on how long the process has been at the current population size.

### 6.4.5 General equilibrium solution for a birth-death process

We now return to computing the equilibrium probability distribution for a general birth-death process, as shown in Equation 26 and Figure 10. We immediately see that

$$P_1 = \frac{\lambda_0}{\mu_1}P_0 \quad \text{(EQ 36)}$$

Substituting this back into Equation 26, we find that

$$P_2 = \frac{\lambda_0\lambda_1}{\mu_1\mu_2}P_0 \quad \text{(EQ 37)}$$

Repeating this substitution, we find that $P_j$ is given by

$$P_j = \frac{\lambda_0\lambda_1\dots\lambda_{j-1}}{\mu_1\mu_2\dots\mu_j}P_0 = P_0\prod_{i=0}^{j-1}\frac{\lambda_i}{\mu_{i+1}} \quad \text{(EQ 38)}$$

We therefore obtain the long-term probabilities of being in any state *j* as a function of the probability of being in state 0 and the system parameters. Knowing that these probabilities sum to 1, we see that

$$P_0 = \frac{1}{1 + \sum_{j=1}^{\infty} \prod_{i=0}^{j-1} \frac{\lambda_i}{\mu_{i+1}}} \qquad \textbf{(EQ 39)}$$

This can be substituted back into Equation 38 to obtain the long-term probability of being in any state *j*. Of course, we need to ensure that the sum of products in the denominator of Equation 39 actually converges! Otherwise, $P_0$ is undefined, and so are all the other $P_i$s. It turns out that the condition for convergence (as well as for the chain to be ergodic) is the existence of a value $j_0$ such that for all values of $j > j_0$, $\lambda_j < \mu_j$. After the population reaches some point, the rate of departures must exceed the rate of arrivals. This makes intuitive sense: otherwise, the population size will grow (in expectation) without bound and the probability of any particular population size will be 0. (You may be wondering whether this applies to a pure-birth Poisson process. The answer is that for such a process, every state is transient, so the chain is not ergodic, and the probability of being in any state is zero).

**Example 19: (General equilibrium solution)**

Find the equilibrium probabilities of being in each state for the birth-death process shown in Figure 11.

*Solution*

From Equation 39, we get

$P_0$ = 1/[1 + 1/5 + (1*5)/(5*8) + (1*5*4)/5*8*10)] = 1/[1+1/5 +5/40+20/400] = 1/[1+1/5+1/8+1/20] = 1/[1+0.2+0.125+0.05] = 1/1.375 = 0.73.

We immediately obtain

$P_1$ = 1/5 $P_0$ = 0.2 * 0.73 =0.146.

$P_2$ = 1/8 $P_0$ = 0.125 * 0.73 =0.09.

$P_3$ = 1/20 $P_0$ = 0.05 * 0.73 = 0.0365.

As a check, note that 0.73 + 0.146 + 0.09 + 0.0365 = 1.0025, which is within the rounding error.

[]

## *6.5 The M/M/1 queue*

We are now in a position to study the famous M/M/1 queue, the simplest non-trivial queueing system. Here, the a/b/c notation, also called Kendall notation, denotes that:

1. (the ‘a’ portion in the notation): the arrival process is ‘Markovian’, i.e, it is a Poisson process with exponentially distributed inter-arrival times
2. (the ‘b’ portion in the notation): the departure process is ‘Markovian’, i.e., it is a Poisson process with exponentially-distributed inter-departure times
3. (the ‘c’ portion in the notation): there is a single server

Extended forms of the notation describe the size of the buffers available (we assume an infinite number), the service discipline (we assume first-come-first-served) and other queueing parameters. However, the three-parameter version of the notation is the one that is commonly used.

The M/M/1 queueing system turns out to be a birth-death Markov process where the arrival rate $\lambda$ and departure rate $\mu$ are state-independent (and therefore independent of the population size). We have already seen that a pure-birth (Poisson) process has exponentially-distributed interarrival times. Using similar reasoning, it can be shown that a birth-death process with a constant and state-independent departure rate has exponentially-distributed inter-departure times and that the departure process is a Poisson process. Hence the equivalence of the birth-death process under study and the M/M/1 queue.

We can study the long-term behavior of the M/M/1 queue by removing state-dependence (the subscript $j$) in the transition rates in the analysis of Section 6.4.5. From Equation 38, we find that:

$$P_j = P_0 \prod_{i=0}^{j-1} \frac{\lambda}{\mu} = P_0 \left(\frac{\lambda}{\mu}\right)^j \qquad j \geq 0 \tag{EQ 40}$$

To obtain $P_0$, we use Equation 39 to obtain

$$P_0 = \frac{1}{\left(1 + \sum_{j=1}^{\infty} \left(\frac{\lambda}{\mu}\right)^j\right)} \tag{EQ 41}$$

When $\lambda < \mu$, the infinite sum in the denominator converges, and the denominator reduces to $\left(\frac{1}{1-\frac{\lambda}{\mu}}\right)$, so that

$$P_0 = 1 - \frac{\lambda}{\mu} \tag{EQ 42}$$

The ratio $\lambda/\mu$ represents the intensity of the arrival rate, as a fraction of the service rate and can be viewed as the utilization of the system. The value is important enough that it deserves its own symbol, $\rho$, which allows us to write Equation 42 as

$$P_0 = 1 - \rho \tag{EQ 43}$$

This equation has the intuitive meaning that the probability that the system is idle ($P_0$) is (1 - utilization). It turns out that this relationship is true for *all* queueing systems whose population size is unbounded.

We now use Equation 40 to obtain

$$P_j = \rho^j (1 - \rho) \tag{EQ 44}$$

Note that this is a geometric distribution.

**Example 20: (M/M/1 queue)**

Consider a link to which packets arrive as a Poisson process at a rate of 300 packets/sec such that the time taken to service a packet is exponentially distributed. Suppose that the mean packet length is 500 bytes, and that the link capacity is 1.5 Mbps. What is the probability that the link's queue has 1, 2 and 10 packets respectively?

*Solution*

The packet length is 500 bytes = 4000 bits, so that the link service rate of 1,500,000 bits/sec = 375 packets/sec. Therefore, the utilization is 300/375 = 0.8. When the link queue has 1 packet, it is in state $j$=2, because one packet is being served at that time. Thus, we need $P_2 = 0.8^2 * 0.2 = 0.128$. For the queue having two packets, we compute $P_3 = 0.8^3 * 0.2 = 0.1$. For 10 packets in the queue, we compute $P_{11} = 0.8^{11} * 0.2 = 0.0067$, an fairly small quantity. Thus, even when the utilization is high (80%), the queue size is quite small, rarely exceeding 10 packets.

[]

Note that the long-term probability that the population size is $j$ depends only on the utilization of the system. As the utilization increases, the probability of reaching larger population sizes increases. To see this analytically, consider the mean number of customers in the system (which is also the mean population size), denoted $\bar{N}$ defined by

$$\bar{N} = \sum_{j=0}^{\infty} jP_j \qquad \textbf{(EQ 45)}$$

It can be shown that this sum converges (when $\lambda < \mu$) and that

$$\text{Mean number of customers in the system} = \bar{N} = \frac{\rho}{(1-\rho)} \qquad \textbf{(EQ 46)}$$

**Example 21: (Mean number of customers in the queue)**

Compute the mean number of packets in the system of Example 20.

*Solution*

The utilization is 0.8, so the mean number of packets in the system is 0.8/(1-0.8) = 0.8/0.2 = 4. Of these, we expect three to be in the queue, and one will be in service.

[]

It is immediately obvious from Equation 46 that as $\rho \to 1$, $\bar{N} \to \infty$. That is, as the arrival rate approaches the service rate, the expected number of customers in the system grows without bound. This is somewhat unexpected: after all, the arrival rate is smaller than the service rate: why then should the number of customers grow? The reason is that we are dealing with stochastic processes. Even though the arrival rate, on average, is lower than the service rate, there will be time periods when the short-term arrival rate exceeds the service rate. For instance, even if the mean arrival rate is 1 customer per second, there will be short intervals during which two or even three customers may arrive in one second. During this time, the queue builds up, to be drained when the service rate exceeds the arrival rate. In fact, there is an interesting asymmetry in the system: when the (short-term) arrival rate exceeds the (short-term) service rate, the queue builds up, but when the service rate exceeds the arrival rate, if the queue is empty, the system does not build up ‘service credits.’ The server is merely idle. Thus, the stem tends to build up queues that are only drained slowly. This is reflected in the fact that as the utilization of the system increases, the mean number of customers in the system increases.

It is remarkable that this fundamental insight into the behavior of a real queueing system can be derived with only elementary queueing theory. Moreover, this insight carries over to all other queueing systems: as the utilization approaches 1, the system becomes *congested*. The behavior of the mean queue length (which also corresponds to the waiting time, through Little's law), is shown in Figure 12.

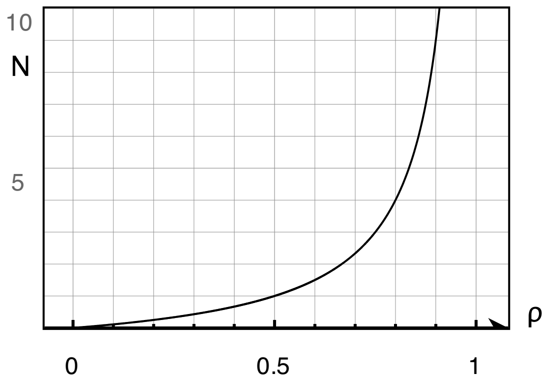


**FIGURE 12.** **Mean queue length as a function of utilization**

It is clear that the queue length asymptotes to infinity as the utilization approaches 1. In networking terms, this means that as the arrival rate approaches a link's capacity, the queue at the immediately preceding router or switch will grow without bound, causing packet loss. This analysis allows us to derive a practical guideline: we should provision enough service capacity so that the system utilization never exceeds something like 70%. Alternatively, if this threshold is exceeded, new service capacity should be made available so that the utilization decreases.

Another related quantity of interest for this queue is the mean waiting time in the queue. From Little's law, the mean number of customers in the system is the product of their mean waiting time and their mean arrival rate, so $\frac{\rho}{(1-\rho)}c$= mean waiting time * $\lambda$, which means that

$$\text{Mean waiting time} = \frac{\frac{\rho}{\lambda}}{(1-\rho)} = \frac{\frac{1}{\mu}}{(1-\rho)} \qquad \textbf{(EQ 47)}$$

This quantity also grows without bound as the utilization approaches 1.

**Example 22: (Waiting time of a M/M/1 queue)**

What is the mean waiting time for a packet in the queue described in Example 20?

*Solution*

For this queue, $\mu$=375 and $\rho$=0.8. So, the mean waiting time is (1/375)/(1-0.8) = 5/375 seconds = 13.3 ms.

[]

## *6.6 Two variations on the M/M/1 queue*

We now consider two variations on the M/M/1 queue, essentially to give insight into how one proceeds with the analysis of a queueing system. The coverage here is necessarily brief, but the steps followed are those that would arise in the analysis of any queueing system.

### 6.6.1 The M/M/$\infty$ queue: a responsive server

Suppose that a provider of service capacity brings on a new server to serve every arriving customer. This would be like a private bank where new agents are brought on to provide individual attention to each customer when she or he arrives. This can be thought of as a queue with an infinite number of servers (though, at any time, the number of servers is finite).

We can model and analyse this queue using the same techniques as an M/M/1 queue. We start with the state-transition-rate diagram show in Figure 13. Note that $\mu_j$, the rate of departure from the $j$th queue, is $j\mu$

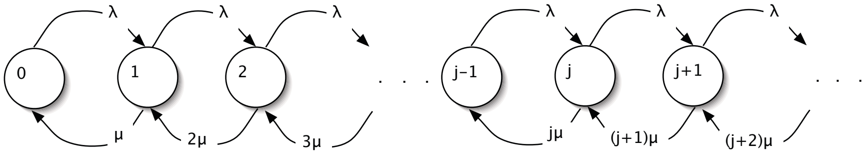


**FIGURE 13. State-transition-rate diagram for an M/M/$\infty$ queues**

which models the fact that when there $j$ customers, there are $j$ servers. From the diagram, we can proceed directly to Equation 38 to write down $P_j$, the probability of being in state $j$, as

$$P_j = P_0 \prod_{i=0}^{j-1} \frac{\lambda}{(i+1)\mu} = P_0 \left(\frac{\lambda}{\mu}\right)^j \frac{1}{j!} \qquad \text{(EQ 48)}$$

Thus, we solve for $P_0$ using Equation 39 as

$$P_0 = \frac{1}{\left[1 + \sum_{j=1}^{\infty} \left(\frac{\lambda}{\mu}\right)^j \frac{1}{j!}\right]} \qquad \text{(EQ 49)}$$

Recalling the standard expansion $e^x = \sum_{j=0}^{\infty} \frac{x^j}{j!}$, we see that

$$P_0 = e^{-\frac{\lambda}{\mu}} \qquad \text{(EQ 50)}$$

$$P_j = P_0 \prod_{i=0}^{j-1} \frac{\lambda}{(i+1)\mu} = e^{-\frac{\lambda}{\mu}} \left(\frac{\lambda}{\mu}\right)^j \frac{1}{j!} \qquad \text{(EQ 51)}$$

Note that Equation 51 is the standard equation for the Poisson distribution! Thus, with ‘infinite’ servers, the number of customers follows the Poisson distribution. This allows us to compute the expected number of customers as mean of the Poisson, which is its parameter, i.e., $\lambda/\mu$. All other parameters of interest for this queueing system can be derived from Equation 51.

**Example 23: (Responsive server)**

Suppose customers arrive at a private bank, modelled as a responsive server, as a Poisson process at the rate of 10 customers/hour. Suppose that a customer’s needs can be met on average in 20 minutes, and that the service time distribution is exponentially distributed. What is the probability that there are five customers in the bank at any point in time?

*Solution*

We have $\lambda = 10$ and $\mu = 3$ (i.e. 3 customers can be served an hour, on average, by a server). Thus $P_0 = e^{-10/3} = 0.036$. We need to find $P_5 = 0.036 * (10/3)^5 * 1/5! = 0.123$. Thus, there is a nearly one in eight chance that there will be 5 customers in the bank at any given time.

[]

### 6.6.2 M/M/1/K: bounded buffers

Suppose that the queueing system has only *K-1* buffers. In this case, the population size (including the customer in service) cannot grow beyond *K,* and arrivals when the system is in state *K* are lost (similar to packet loss when arriving to a full queue). To model this, we can simply ignore arrivals to state *K*, which means that we will never enter states *K+1, K+2,...* This results in a state-transition-rate diagram shown in Figure 14.

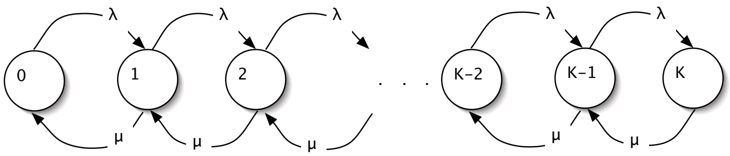


**FIGURE 14. State-transition-rate diagram for an M/M/1/K queue**

The state transition rates are therefore

$$\lambda_j = \begin{cases} \lambda & j < K \\ 0 & j \geq K \end{cases} \qquad \text{(EQ 52)}$$

$$\mu_j = \mu \qquad \text{j=1,2,...,K}$$

We can therefore use Equation 38 to write down $P_j$ as

$$P_j = \begin{cases} P_0\left(\frac{\lambda}{\mu}\right)^j & j \leq K \\ 0 & j > K \end{cases} \qquad \text{(EQ 53)}$$

We can substitute this in Equation 39 to get

$$P_0 = \frac{1}{\left[1 + \sum_{j=1}^{K}\left(\frac{\lambda}{\mu}\right)^j\right]} \qquad \text{(EQ 54)}$$

Given the standard result $\sum_{k=0}^{n-1} r^k = \frac{1-r^n}{1-r}$, we can simplify this to

$$P_0 = \frac{1-\frac{\lambda}{\mu}}{1-\left(\frac{\lambda}{\mu}\right)^{K+1}} \qquad \text{(EQ 55)}$$

So, we can now write Equation 53 as

$$P_j = \begin{cases} \dfrac{1-\dfrac{\lambda}{\mu}}{1-\dfrac{\lambda}{\mu}^{K+1}}\left(\dfrac{\lambda}{\mu}\right)^j = \dfrac{1-\rho}{1-\rho^{K+1}}\rho^j & j \le K \\ 0 & j > K \end{cases} \tag{EQ 56}$$

As before, given these probabilities, we can compute all quantities of interest about the queueing system, such as the distribution of the queue length, the mean number of customers in the queue, and the mean waiting time. In particular, the intuitive meaning of $P_K$ is the probability that the system is 'full' when it has a buffer of size *K-1*. So, $P_K$ can be interpreted as the *blocking probability* of a M/M/1/K queue. We can then choose *K* as a sizing parameter to make $P_K$ as small as desired.

Note that in this system, $P_0 \neq 1 - \rho$ because the system size is bounded (specifically, the number of customers served in a chosen time period may be lower than what the utilization indicates because customer arrivals when the queue is full are lost). Moreover, the system is stable by definition, independent of the utilization, because excess arrivals are automatically dropped.

**Example 24: (M/M/1/K queue)**

Consider the system of Example 20, but with the restriction that the queue only has four buffers. What is the probability that three of these are in use? How many buffers should we provision to ensure that the blocking probability is no more than $10^{-6}$?

*Solution*

We have $K = 5$, and $\frac{\lambda}{\mu}$=0.8. From Equation 55, we get $P_0$ = (1-0.8)/(1-$0.8^6$) = 0.27. If three buffers are in use, then the system is in state $j$=4. From Equation 53, we get $P_4$ = 0.27$(0.8)^4$=0.11.

To size the buffer, we have to choose $K$ such that $P_K < 10^{-6}$. We solve for $K^*$ in $10^{-6}$> ((.2)$(0.8)^K$)/(1-$0.8^{K+1}$*),* so $K^* = 55$. Thus, we need 54 buffers to satisfy this blocking probability.

[]

These two examples should give some insight into the modelling and analysis of birth-death systems. The literature has many other examples of such systems, but we will not consider them further.

## *6.7 Other queueing systems*

We now turn our attention to queueing systems that go beyond the Markovian and exponential framework. The queueing systems become much harder to analyze in this case, so we will merely state the results for two important queueing systems.

### 6.7.1 M/D/1: deterministic service times

Consider a queueing system where arrivals are from a Poisson process, but service times are deterministic. That is, as long as the queue is non-empty, the inter-departure time is deterministic (rather than exponentially distributed). Representing the inter-departure time (a constant) by $\mu$, and the utilization by $\rho=\lambda/\mu$, it can be shown that the system is stable (i.e., the queue length is finite) as long as $\lambda < \mu$. Moreover, the long-term probability that the number of customers in the system is $j$, i.e., $P_j$ is given by

$$P_j = \begin{cases} 1-\rho & j=0 \\ (1-\rho)(e^{\rho}-1) & j=1 \\ (1-\rho)\left(\sum_{i=0}^{j} \frac{(-1)^{j-i}(i\rho)^{j-i-1}(i\rho+j-i)e^{i\rho}}{(j-1)!}\right) & j>1 \end{cases} \qquad \text{(EQ 57)}$$

This allows us to derive the mean number of customers in the system as:

$$\text{Mean customers in the system} = \rho + \frac{\rho^2}{2(1-\rho)} \qquad \text{(EQ 58)}$$

and the mean response time as:

$$\text{Mean response time} = \frac{1}{\mu} + \frac{\rho}{2\mu(1-\rho)} \qquad \text{(EQ 59)}$$

Other quantities of interest regarding the M/D/1 queue can be found in standard texts on queueing theory, such as [Kleinrock Vol. 1].

### 6.7.2 G/G/1

Once the arrival and service processes become non-Poisson, the analysis of even a single queue becomes challenging. For such systems few results are available other than Little's law, and also that, if the queue size is unbounded, $P_0 = 1\text{-}\rho$. A detailed study of such queues is beyond the scope of this text.

### 6.7.3 Networks of queues

So far, we have only studied the behavior of a single queue. This is like studying a network with a single router - not very interesting! What happens when we link the output of a queue to the input of another queue, as we do in any computer network? Intuitively, what we are doing is to make the inter-departure process of one queue the inter-arrival process for the second queue. Moreover, we may have more than one inter-departure process mix to form the inter-arrival process. Can this be analyzed?

We represent this composite system, also called a 'tandem of queues' as shown in Figure 15.

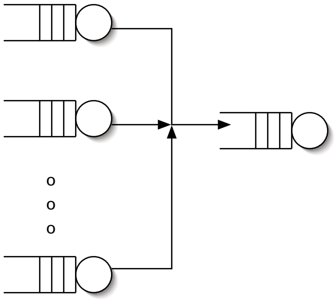


**FIGURE 15.** A network of queues

Here, each queue is shown by a buffer (with customers or jobs in it) and a server (represented by a circle). Jobs served by the servers on the left enter the queue of the server on the right. Each queue and associated server is also called a *node* (drawing on the obvious graph analogy).

If all the queues on the left are M/M/1 queues, recall that their inter-departure processes are Poisson. Moreover, it can be shown that the mixture of Poisson processes is also a Poisson process whose parameter is the sum of the individual processes. Therefore, the input to the queue on the right is a Poisson process, which means that we can analyze it as an M/M/1 queue. This leads to the fundamental insight that a tandem of M/M/1 queues is analytically tractable. Because the departure process of a M/M/m queue (i.e. a queue with m servers) is also Poisson, this result holds true for tandems of M/M/m queues.

We can make things a bit more complicated: we can allow customers to enter *any* queue (node) as a Poisson process and we can also allow customers that leave a node to exit the system altogether with some probability or join any other node in the system with some probability. Note that this can potentially lead to cycles, where customers go through some set of nodes more than once. Nevertheless, Jackson was able to show that these networks behave as if each M/M/m queue was being fed by a single Poisson stream. Such networks are also called *Jacksonian* networks in his honour. For a Jacksonian network, we have a strong result: let $P_{k1k2k3..kn}$ denote the long-term probability that there are *k1* customers at the first node, *k2* customers at the second node, and so on. Then

$$P_{k1k2k3...kn} = P_{k1}P_{k2}P_{k3}...P_{kn} \tag{EQ 60}$$

That is, the joint probability of having a certain number of customers in each queue is just the product of the individual probabilities, which means that queues act as if they are independent of each other. This *product-form* of the probability distribution greatly simplifies analysis. However, we will not explore this further here.

Despite the elegance and power of Jacksonian network analysis, these results rarely apply in practical computer networks. This is because customers (packets) rarely arrive as a Poisson process. Thus, the output process is also non-Poisson, which makes subsequent analysis complex. In recent years, the development of network calculus and stochastic network calculus has allowed significant inroads into the study of non-Jacksonian networks: the interested student should refer to [LeBoudec Thiran].

## 6.8 Exercises

### 1 Little's law

Patients arriving to the emergency room at the Grand River Hospital have a mean waiting time of three hours. It has been found that, averaged over the period of a day, that patients arrive at the rate of one every five minutes. (a) How many patients are awaiting treatment on average at any given point in time? (b) What should be the size of the waiting room so that it can accommodate everyone?

### 2 A stochastic process

Consider that in Example 4, a person is on an infinite staircase on stair number 10 at time 0 and potentially moves once every clock tick. Suppose that he moves from stair *i* to stair *i+1* with probability 0.2, and from stair *i* to stair *i-1* with probability 0.2 (the probability of staying on stair *i* is 0.6). Compute the probability that the person is on each stair at time 1 (after the first move), time 2, and time 3.

### 3 Discrete and continuous state and time processes

Come with your own examples for all four combinations of discrete state/discrete time/continuous state/continuous time processes.

### 4 Markov process

Is the process in Exercise 2 a Markov process? Why or why not?

### 5 Homogeneity

Is the process in Exercise 2 homogeneous? Why or why not?

### 6 Representation

(a) Represent the process in Exercise 2 using a transition matrix and a state transition diagram. (b) Do the rows in this matrix have to sum to 1? Do the columns in this matrix have to sum to 1? Why or why not? (c) Now, assume that the staircase has only 4 steps. Make appropriate assumptions (what are these?) to represent this finite process as a transition matrix and a state transition diagram.

### 7 Reducibility

Is the chain in Exercise 2 reducible? Why or why not?

### 8 Recurrence

Is state 1 in the chain in Exercise 6(c) recurrent? Compute $f_1^1$, $f_1^2$and $f_1^3$.

### 9 Periodicity

Is the chain in Exercise 2 periodic? If not, give an example of a chain with period $N$ for arbitrary $N > 1$.

### 10 Ergodicity

Is any state in the chain of Exercise 6(c) non-ergodic? Why or why not?

### 11 Stationary probability

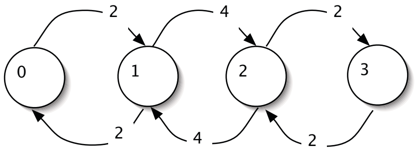
Compute the stationary probability distribution of the chain in Exercise 6(c).

### 12 Residence times

Compute the residence time in each state of the Markov chain in Exercise 6(c).

### 13 Stationary probability of a birth-death-process

Consider the state-rate-transition diagram shown below.



(a) Compare this with the state transition probability diagram in Exercise 6(c). What features are the same, and what differ?

(b) Write down the $\boldsymbol{Q}$ matrix for this system.

(c) Use the $\boldsymbol{Q}$ matrix to compute the stationary probability distribution of this chain.

### 14 Poisson process

Prove that the inter-departure time of a pure-death process is exponentially distributed.

### 15 Stationary probabilities of a birth-death process

Use Equation 30 to compute the stationary probability of the birth-death process in Exercise 13.

### 16 M/M/1 queue

Is the birth-death-process in Exercise 13 M/M/1? Why or why not?

### 17 M/M/1 queue

Consider a link to which packets arrive as a Poisson process at a rate of 450 packets/sec such that the time taken to service a packet is exponentially distributed. Suppose that the mean packet length is 250 bytes, and that the link capacity is 1 Mbps.

(a) What is the probability that the link's queue has 1, 2 and 10 packets respectively?

(b) What is the mean number of packets in the system? What is the mean number in the queue?

(c) What is the mean waiting time?

### 18 Responsive (M/M/$\infty$) server

Compute the ratio of $P_j$ for a responsive server to the same value for an M/M/1 queue. How does this ratio behave as a function of $j$?

### 19 M/M/1/K server

Assume that the queueing system in Exercise 17 has 10 buffers. Compute an upper bound on the probability of packet loss.

## 20 M/D/1 queue

Compute the mean number of customers in an M/D/1 system that has a utilization of 0.9. (a) How does this compare with a similarly loaded M/M/1 system? (b) Compute the ratio of the mean number of customers as a function of $\rho$. (c) Use this to compare the behavior of an M/D/1 queue with that of an M/M/1 queue under heavy load.

# CHAPTER 7 *Game-theoretic concepts*

Mathematical game theory is the study of the behavior of decision-makers who are conscious that their actions affect each other and who may have imperfect knowledge both of each other and of the future. It originated with the mathematical study of 'parlor games' like bridge, chess, and Matching Pennies. In these games, each player makes moves in response to, and in active consideration of, the moves of the other players. The concepts that arise in the study of such parlor games are widely applicable. In particular, they are relevant in situations where a scarce resource has to be shared amongst many entities and each entity tries to maximize its own share: a situation that frequently occurs in computer networks.

The origin of game theory, due almost entirely to von Neumann and Morgenstern in the first half of the twentieth century, was part of a larger project on the mathematicization of sociology which came from the world view that mathematics could be used to solve problems of society that had eluded centuries of past effort by qualitative sociologists. This world view, especially the use of game theory to model policies for global nuclear warfare, gave the theory a (perhaps deservedly so) bad reputation, though it was routinely used to study microeconomic problems. In recent years, game theory has given deep insights into the operation of the Internet and in the design of decentralized algorithms for resource sharing, in particular, the theory known as 'mechanism design.' Hence, there has been a resurgence of interest in these topics. This chapter describes the terminology of game theory, focuses on algorithmic aspects of mechanism design, and concludes with a sketch of the limitations of this approach.

## *7.1 Concepts and terminology*

### 7.1.1 Preferences and preference ordering

The ultimate basis of game theory is utility theory, which in turn is grounded in the axiomatization of the preference relationships of a person. Note that the axioms of preferences refer to 'goods.' By a good, we mean concrete objects such as a bar of soap or a meal at a restaurant, as well as more abstract quantities such as the mean end-to-end delay, measured over intervals of one second and over the course of a given hour, between a particular browser and a web server.

Here are the axioms of preferences:

1. *Orderability:* Given two goods, the person must prefer one to the other or view them both as being equally preferable. There is no option to 'pass.' Therefore, in any set of goods, there must exist a set of equivalent most-preferred and least-preferred goods.
2. *Transitivity*: If a person prefers good A to good B and good B to good C, then they prefer good A to good C.
3. *Continuity*: We assume that if a person prefers good B more than good A and less than good C, it is always possible to define a lottery where with probability *p* the user would win prize A and with the remaining probability the user would win prize C, such that the person equally prefers B and the lottery. We say that the person is indifferent between B and a lottery with outcome *p*A + (1-*p*)C.
4. *Substitutability*: If a person prefers good A and B equally, then we should be able to replace one with the other in any lottery.
5. *Monotonicity*: Given two lotteries with the same outcomes A and C, defined by *p*A + (1-*p*)C, *q*A + (1-*q*)C respectively, if a person prefers A to C and *p*>*q*, then it prefers the first lottery to the second and *vice versa.*
6. *Decomposability*: A *compound lottery* is a lottery that is run in two stages, where the winners of the first stage enter a second lottery and may subsequently either win again or lose. Decomposability means that such a compound lottery is equivalent to an appropriately-defined single-stage lottery: if the outcome of the first lottery is *p*A+ (1-*p*)B, and of the second lottery is *q*C+(1-*q*)D, and outcome B of the first lottery is participation in the second, then the outcome of the compound lottery is *p*A +(1-*p*)*q*C + (1-*p*)(1-*q*)D.

**Example 1: (Preferences)**

Consider a person who prefers an apple (A) to a banana (B) and a banana to a carrot (C). We offer the user a lottery that goes as follows: we divide the circumference of a circle into two sections, marked A and C, where the fraction of the circumference that is marked A is denoted *p*. We then spin a pointer pinned to the centre of the circle. If the pointer stops spinning at a part of the circle marked A (which happens with probability *p*), we give the person an apple. Otherwise, we give them a carrot. The assumption is that there is some value of *p* where the person is equally happy with a banana and the results of this lottery. Intuitively, when *p* is 1, then the person always gets an apple, so the person should prefer the lottery to the banana. Conversely, when *p* is 0, then the person always gets a carrot, so the person should prefer a banana to the lottery. Therefore, it seems plausible, and is an axiom of utility theory, that there is some intermediate point where the person equally prefers the lottery and the banana.

[]

These axioms of preference allows us to express the preference a person may have for any member of a set of goods as a lottery over the preferences for the least and most preferred element. We can do more: suppose that we assign numerical values to the least and most preferred element, say 0 and 1. Then, we can assign the numerical value *p* to the preference of a good *G,* where a person equally prefers *G* and a lottery where they win the most preferred element with probability *p* and the least preferred element with probability 1-*p*. We can therefore think of *p* as being a numerical value for the preference for *G.*

More generally, a *utility function* that assigns numerical values to preferences over goods allows us to numerically compare the preference expressed by a user among these goods. For instance, if we denote the least preferred good by *L* and the most preferred by *M*, then we can define a utility function *U* by: *U(L) = 0, U(M) = 1,* and for all other goods *G, U(G) = p*, where the user equally prefers *G* and a lottery amongst *L* and *M* with odds *p*. With a slight change of perspective, we can imagine that *U* assigns *utilities* to goods, such that higher utilities correspond to more preferred goods (this assumes that a user's preferences are consistent).

Utilities are useful for modelling competing objectives in multi-objective optimization. Suppose we wish to optimize a system where there is an inherent conflict amongst the objectives. For example, in most systems, increased performance comes at increased cost. We desire better performance, but we also desire lower cost. We can model these competing objectives with a utility function that increases with performance and decreases with cost (where the cost, itself, may depend on the performance). This naturally models the preferences we have over the 'goods' of performance and cost. By maximizing this utility function, we can find the choice of system parameters that makes the most desirable trade-off between performance and cost.

**Example 2: (Utility function)**

Consider a network user who wants to transfer large files using an ISP that charges per-GB for transfer above a monthly quota, say $1.25/GB. Obviously the user would like to transfer unlimited amounts of data, but also would not like to pay a large monthly bill. We can model this using a utility function. Specifically, assume that a user prefers more data transfer to less data transfer and smaller monthly payments to larger monthly payments. Let $d(x)$ denote the utility of $x$GB of transfer, where $d$ is an increasing function of $x$, and let $p(y)$ denote the (dis-) utility of $y$ dollars of payment, where $p$ is an increasing function of $y$. Of course, $y$ itself is a function of $x$, so we can write it as $y(x)$. The overall utility, $U$ increases with $d$ and decreases with $p$, modelling the conflict in the underlying objectives.

A typical form assumed for $U$ is $U = ad(x) - p(y(x))$, where $a$ is a tuning parameter that is chosen to balance the relative utilities of data transfer and money. Setting $U$ to 0, we see that $a$ is just $p(y(x))/d(x)$. That is, we can determine $a$ by finding the amount of data transfer at which the cost of transfer just balances the gain. Of course, $U$ can be far more complicated. Note also that $U$ is linear function of $d$ and $p$. If $d$, $p$, and $y$ are also linear functions, $U$ would be a linear function of $x$. This is another sense in which $U$ can be linear (compare it with the earlier definition of linearity).

Finally, it is worth observing that linear utility functions in this sense are rather unrealistic: most people experience diminishing returns with increases in the quantity of a good, which is better modelled by a non-linear curve of the form $1-e^{-x}$.

[]

Before we leave the topic of utility functions, we remark on two important properties. First, utilities are unique only up to an affine transform. That is, the preference orderings established by utility functions $U_1$ and $aU_1+b$, where $a$ and $b$ are real constants, are identical. Indeed, we have *no* way to distinguish amongst them. This brings us to the second important property of utility functions: they are *personal*. Your utility function and mine are incomparable in a deep sense: what you prefer and what I prefer just cannot be compared. So, if I were to assign a utility of 5 to some good, and you were to assign a utility of 7 to the same good, it does *not* mean that you prefer it more. I could easily have assigned it a utility of 5000, for example, with an affine translation of my utility function. Therefore, any scheme that requires interpersonal utility comparison is simply wrong.

We will now examine some elementary concepts of game theory.

### 7.1.2 Games, players, and actions

Game: Every game involves interaction between players. The outcome of the game depends on their actions. Moreover, each player chooses their action in response to prior actions by other players, as well as expectations about how their action will ‘play out’ in the other player’s minds. This is what makes analyzing a game both complex and interesting.

Perhaps the difference between a game and mathematical optimization can be intuited from an example. Suppose you are looking for a particular store in a mall in an unfamiliar country where you cannot decipher the signs. This is an optimization problem, where you choose some search strategy that minimizes the time taken to find the store, given the limited set of actions you have (walk past every store, ask for help, etc.). In contrast, suppose you are looking for your lost friend in a mall. Should you stay in a central location, so that your friend can find you? Or should you move around, with some risk that you may never meet? Introducing a second locus of decision making completely changes the problem!

Players, actions, and payoffs: We model the interaction of active decision-makers in the form of a *game*, where each decision-maker or *player* takes *actions* chosen from a finite or infinite set. These actions may be simultaneous or sequential, and there may be only one action or many. When the game ends, that is, all the actions are carried out, each player is given a reward (also called an *outcome*). We assume that each player has a utility function that establishes a preference ordering among the outcomes--the utility of an outcome is also called the *payoff*.

Rationality: We assume players are *rational*, that is, they make actions that will assure them the best possible outcome. The *Principle of Maximum Expected Utility* states that a rational player takes actions that maximize its expected utility. Moreover, player rationality is assumed to be common knowledge: each player knows that every other player is rational, and this fact itself is known by everyone, with infinite recursion. This is one of the most controversial assumptions made by game theory.

Non-cooperation: We will assume in the discussion here that players do not cooperate or form coalitions: it is every player for themselves (non-cooperation, however, is not the same as adversarial or malicious behavior). This holds true for almost all game-theoretic models of computer networking problems. The area of cooperative game theory is rich, but not something we will touch upon.

Information: A critical factor in distinguishing among games is the degree of information each player has about other players and about the game. In some games, players may be able to observe every action and may also precisely know what every player's possible actions and objectives are (these are games of *perfect information*). A prototypical example is the game of chess, where all possible actions of each player are codified by the rules of chess, and all actions are visible to the players. In other games, players may not be able to see other players' actions, not know other players' objectives, and, in some cases, may not even know how many other players there are in the first place. For example, in the game of bridge, each player's hand is private, so a player cannot know the set of potential next actions. These limitations, naturally, limit the degree to which we can model the games and predict their outcomes. In some cases, such as when a player does not know the initial state of the game (for instance, any game of cards where the deck has been shuffled so no player knows what the other players' hands contain), we can think of there being a special player called 'Nature' that makes a randomizing initial move, after which the players play the actual game.

### 7.1.3 Strategies

A player's actions typically depend not only on prior actions of the other players (the *history* of the game) but also expectations on the responses this action may elicit from the other players. A player's *strategy* is a full description of the player's actions during the entire game. A strategy should be detailed enough that a player can give it to a disinterested third party and walk away (or give it to a computer for execution). It describes the player's actions taking into account every possible action by every other player. Clearly, a full description of a strategy is impossible other than for the simple games. Nevertheless, the concept of a strategy is both useful (in that it allows us to precisely model a player's actions) and necessary (to determine the expected outcome of a game, which is expressed in the form of each player's preferred strategy). In every game-theoretic analysis, a critical modelling step is to enumerate a small set of strategies for each player.

There are two types of strategies. *Pure* strategies deterministically describe which actions a player makes at each stage of the game. In contrast, *mixed* strategies associate probabilities with two or more pure strategies. The actual strategy that is played is chosen according to these probabilities. In this sense, a pure strategy is a mixed strategy with the entire probability mass at a single point in the domain. Note that with a mixed strategy, once a specific pure strategy is chosen, the player cannot introduce any additional randomness--every action must be made deterministically in accordance with the pure strategy. In a repeated game, where a game is played repeatedly, a probabilistic choice can be made for each underlying game instance.

**Example 3: (Pure and mixed strategies)**

Consider the simple game of 'Matching Pennies.' In this game, two players each have a coin that they simultaneously place on a table. Subsequently, each observes both coins. One player wins if the coins are both heads or both tails and the other player wins otherwise. A player has two pure strategies: play heads or play tails. However, we can also define an infinite number of mixed strategies, parametrized by a real number $p$ in [0,1], each of which corresponds to the strategy: play heads with probability $p$ and tails with probability 1-$p$. We will show later that the optimal strategy for both players is to choose $p$=0.5.

[]

We denote the strategy adopted by player $i$ by $s_i$. Thus, we can denote an entire game played by $n$ players by the tuple of adopted strategies $(s_1, s_2,...,s_n)$, which is also called the *strategy profile*. The payoff to player $i$ as a result of the game is denoted $\pi_i(s_1, s_2,...,s_n)$ and represents the utility of player $i$ when the game is played with that particular strategy profile.

**Example 4: (Strategy profile)**

For the game of Matching Pennies, suppose we denote the action 'play head' by H and the action 'play tail' by T. Then, the set of possible pure strategies is {HH, HT, TH, TT}, where each element is a strategy profile. Suppose that for both players, winning has utility 1 and losing has utility -1. Also, let player 1 win if the coins match and player 2 win otherwise. Then, we can write down the payoffs as:

$\pi_1(HH) = 1, \pi_1(HT) = -1, \pi_1(TH) = -1, \pi_1(TT) = 1;$

$\pi_2(HH) = -1, \pi_2(HT) = 1, \pi_2(TH) = 1, \pi_2(TT) = -1;$

Note that in this example, the utility of player 2, for each strategy, is exactly the opposite of player 1. In other words, player 1 wins if and only if player 2 loses. Such a game is called a *zero-sum* game.

[]

### 7.1.4 Normal- and extensive-form games

Games can be represented in two standard forms: *normal form* and *extensive form*. In normal form, a game with *N* players is represented by an *N*-dimensional array, where each dimension corresponds to the possible pure strategies of each player and each matrix element is an *N*-tuple that corresponds to the outcome for each player; the *i*th element of each tuple is the payoff (utility) to the *i*th player. This array is also called the *payoff matrix*. Note that all the strategies are assumed to be played simultaneously.

There is a quirk with representing games in normal form that the following example demonstrates. Consider a two-person game where Row can play pure strategy A or B, and if it plays A, it wins and the game is over. On the other hand, if it plays B, Column can play Y or N, and Row can then play C or D. What is Row's strategy space? By convention, it is not {A, BC, BD}, but {AC, AD, BC, BD}. That is, even though the game ends after Row plays A, we represent the alternatives AC and AD explicitly in the normal form.

**Example 5: (Normal form: Matching Pennies)**

Here is the Matching Pennies game in normal form, where player 1's pure strategies are along the rows, and player 2's pure strategies are along the columns:

| | **H** | **T** |
|---|---|---|
| **H** | (1,-1) | (-1,1) |
| **T** | (-1,1) | (1,-1) |

TABLE 1. **The payoff matrix for Matching Pennies**

[]

**Example 6: (Normal form: WiFi game)**

In the 802.11 (WiFi) protocol, each station with data to send contends for airtime. For simplicity, assume that time is slotted and that each packet transmission takes exactly one time slot. If a station does not send data in a slot, airtime is wasted. If only one station sends data, it succeeds. If both send, both fail, and both must try again.

Consider an 802.11 wireless LAN with two active stations. We can model this as a game with two players. Consider actions over a single time slot. For each player, the possible actions are 'send' (S) and 'don't send' (D). We need to model the payoffs to each player as well. We assume that a station prefers success to a wasted time slot, and prefers a wasted time slot to a collision, because on a collision, not only is no progress achieved, but energy is also wasted. We can express these preferences by assigning a utility of -1 to a collision, 0 to a wasted time slot, and 1 to success. This allows us to represent the game in normal form as shown below:

| | **S** | **D** |
|---|---|---|
| **S** | (-1,-1) | (1,0) |
| **D** | (0,1) | (0,0) |

TABLE 2. **The payoff matrix for the WiFi game.**

[]

In extensive form, the game is represented by a *game tree* where each node corresponds to the player whose turn it is to move and each departing edge represents a possible action that can be taken at that stage of the game. We can identify each node in the tree with the past *history* of the game, i.e., the previous actions taken by each player to get to that stage of the game. Leaves correspond to outcomes of the game and are associated with the payoff to the players for that outcome. Games with sequential actions by the players are more naturally represented in extensive form.

**Example 7: (Extensive form: the Pricing game)**

Suppose an Internet Service Provider wants to roll out a new service with a price that could be low (L), medium (M) or high (H). Suppose the prices are 1, 2, and 3 respectively and the utility that the customer derives from the service is *a*. For each price, the customer can say yes (Y) or no (N). If the customer says yes, the ISP gets a payoff corresponding to the price and the customer gets the payoff *a - price*. If the customer says no, the payoff to both parties is 0. In extensive form, we represent it as shown in Figure 16.

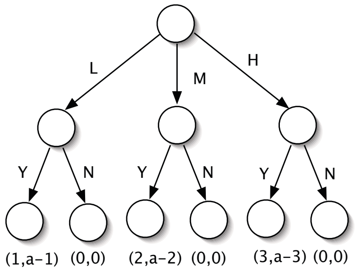


FIGURE 16. **Extensive form of the pricing game**

[]

The extensive-form representation has an inherent problem with representing simultaneity. Consider a game where two players make simultaneous actions, such as Matching Pennies. If we arbitrarily choose nodes corresponding to player 1's actions as the first level of nodes, then, when player 2 takes an action, it can see what player 1 did. This, of course, makes the game pointless. Making the alternative choice does not remedy the situation. What we need is a way to hide a player's actions from the other player, that is, make the information available to it less than 'perfect.' We can do so by placing a subset of nodes in the game tree in the same *information set*. From the perspective of a player, all nodes in the set are equivalent (the player cannot distinguish between them). Graphically, we draw a dashed line between equivalent nodes. This allows us to easily represent simultaneous actions, as the next example shows.

**Example 8: (Representing simultaneous actions in extensive form)**

Figure 17 shows the extensive-form representation of the Matching Pennies game. Note that Player 2 cannot make out whether Player 1 played H or T.

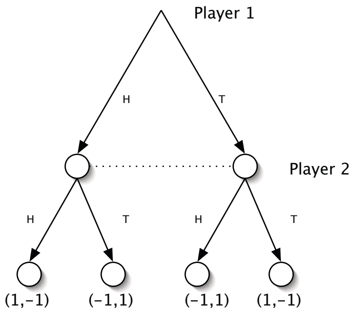


FIGURE 17. **Representing simultaneity in an extensive-form game**

[]

It can be shown that, with the use of information sets, the normal and extensive form representations are equivalent. Therefore, from now on, we will not distinguish between the two.

Both normal and extensive forms grow exponentially in size with the number of actions and number of players. If there is only limited interaction between the players, we can use a compact *graphical game* representation instead.

A graphical game with *n* players is represented by an undirected graph with *n* vertices and a set of *n* payoff matrices, with one vertex and one matrix corresponding to each player. If node *i* has neighbours *j* and *k*, then player *i*'s payoffs are only a function of *i*, *j*, and *k*'s actions, and therefore the payoff matrix at node *i* only has entries for *i, j*, and *k*'s actions. The overall game is therefore composed from the individual sub-games. Note that we are not ruling out global impact of a player's actions. However, this occurs only through the propagation of local influences. This is similar in principle to Bayesian networks, where the joint probability distribution over all the underlying random variables is a product of the local conditional probability distributions at each node, and where the absence of edges between two vertices denotes independence. Kearns et al [M. Kearns, M. Littman, and S. Singh, "Graphical models for game theory," Proceedings of the Conference on Uncertainty in Artificial Intelligence, 2001] proposed this model in 2001 and it is rapidly gaining popularity because it can represent games that cannot be compactly represented in normal or extensive form. Note that a graphical game is equivalent to a normal form game: it is just a compact representation that leaves out inessential parts of the matrix.

## 7.1.5 Response and best response

Consider a two-player game with players labelled A and B. Recall that A's strategy $s_A$ encapsulates this player's actions for the entire game. The strategy chosen by B *conditional on A playing* $s_A$ is called B's response to $s_A$. Now, given the set of possible strategies that B could play, at least one of them will have a payoff as good as or higher than all the others. We call this B's *best response* to $s_A$.

We can generalize this to *n* players as follows. Let $(s_1, s_2,...,s_n)$ be a strategy profile. We denote by $s_{-i}$ the tuple $(s_1, s_2,...,s_{i-1}, s_{i+1},...,s_n)$, that is, the strategy profile excluding *i*'s strategy. Then, the best response of player *i* to $s_{-i}$ denoted $s_i^*$ are the strategies (there may be more than one) that give *i* the highest possible payoff. That is:

$$\pi_i(s^*_i, s_{-i}) \geq \pi_i(s^j_i, s_{-i}) \qquad \forall s^j_i \neq s^*_i \tag{EQ 61}$$

**Example 9: (Best response)**

Consider the one-step WiFi game in Example 6. If the row player (station 1) plays S, then the column player's (station 2's) best response is D, because the payoff for SD is 0, and for SS is -1. On the other hand, if the row player plays D, then the best response is S, because the payoff for DS is 1, and for DD is 0.

[]

## 7.1.6 Dominant and dominated strategy

In some games, it is possible to determine that a rational player should play a particular strategy *no matter what* all the others players play. This is called a *dominant strategy* for that player. Mathematically, we say that a strategy $s_i^*$ is a dominant strategy for player *i* if

$$\pi_i(s^*_i, s_{-i}) \geq \pi_i(s^j_i, s_{-i}) \qquad \forall s_{-i}, \forall s^j_i \neq s^*_i \tag{EQ 62}$$

Compared to the best response, there is an additional universal quantifier over all $s_{-i}$, which indicates that the dominant strategy is the best response no matter what strategies the other players pick. If the inequality is strict, then the strategy is *strongly* dominant, else, if it is strict for at least one $s_{-i}$ but not for all of them, it is *weakly* dominant.

Symmetrically, a strategy whose payoff is lower (strictly lower) than another strategy is weakly (strictly) dominated by the other strategy.

**Example 10: (Dominant strategy)**

In the WiFi game of Example 6, the column player's best response is S or D, depending on what the row player does. Therefore, this game does not have a dominant strategy for the column player. Symmetrically, the game also does not have a dominant strategy for the row player.

Wireless networks can exhibit the *capture effect*, where a transmission from a station succeeds even if there are competing transmissions, because the signal strength from that station is so strong that it overpowers the competition. In this case, the payoff matrix is:

| | **S** | **D** |
|---|---|---|
| **S** | (-1,1) | (1,0) |
| **D** | (0,1) | (0,0) |

TABLE 3. **Payoff matrix for WiFi game with capture effect.**

Transmissions from the column player (station) always succeed due to the capture effect. In this case, the dominant strategy for the column player is S: no matter what the row player does, the column player is better off doing S than D.

[]

### 7.1.7 Bayesian games

We stated earlier that a critical factor in any game is the amount of information available to each player. In a game with perfect information, each player knows the other players, their past history of actions, the actions available to the other players, and the utility to the other players from each outcome. Consider a card game which starts with the deck being shuffled and dealt. No player knows the other players' hands, so they cannot know their possible actions. This is, therefore, a game with *imperfect* information.

A *Bayesian* game is a form of game with imperfect information, where each player can be from a set of possible *types*. If all players know each others' types, then the game becomes one of perfect information. Therefore, all the uncertainty is encapsulated in the selection of player types. In a card game, each player's hand corresponds to that person's type. The *state* of the game is the collection of player types for that game.

A Bayesian game starts with a move by Nature that results in each player being randomly assigned a type, according to some probability mass function. We can also view this as Nature selecting some state of the game. At this point, it is possible that all players receive a *signal* that may give information about each other. Specifically, the signal eliminates some possible states. Thus, the conditional probability that a player has a particular type, given a signal, is greater than the unconditional probability of that type.

In an extensive-form game, imperfectness can be represented by putting more than one node in an equivalence class, where a player is unable to distinguish between nodes in the same equivalence class (they are in the same information set). The effect of a signal is to potentially reduce the size of an equivalence class.

**Example 11: (Bayesian WiFi game)**

Consider the 802.11 game of Example 6, where the row player does not know the column player's signal strength. If the column player's signal strength is low, then the capture effect is absent, and the payoff matrix is the one shown in Table 2 on page 143. Otherwise, with the capture effect, the payoff matrix is shown in Table 3 on page 146. We can model this as a Bayesian game where the column player has one of two types: 'strong signal' and 'weak signal'. In the first move by Nature, the column player is assigned one of the two types according to some probability distribution (say, with probability 0.1 the player is strong, and with probability 0.9 the player is weak).

To incorporate the notion of a signal, consider that the row player can measure the received signal strength of the column player's transmission. Suppose that, given that the signal strength is high, the column player has 0.95 probability of being strong and 0.05 probability of being weak. Similarly, assume that the signal strength is low, the column player has a 0.1 probability of being strong and 0.9 probability of being weak. We can see that the signal allows the row player to better judge the type of the column player, potentially allowing it to improve its chances in the game.

[]

### 7.1.8 Repeated games

Nearly any game can be repeated. Each repetition of a game is called a *stage*. With a finite number of repetitions, the overall game can be thought of as a single game with a much larger strategy profile space because each player can change strategies at each repetition, perhaps taking into account the outcome of the previous stage. Analysis of finite repeated games is thus more complex and does not necessarily give much more insight into the problem.

Paradoxically, the analysis of repeated games is somewhat simpler when games are repeated an infinite number of times. The reason is that in such a case every stage is equivalent: we do not have a 'final' stage that needs to be specially handled. In a game with infinite repetitions, both the normal and extensive forms are undefined. The payoff matrix in the normal form is infinite-dimensional, which means that we cannot represent the payoffs. Similarly, in the extensive form, there are no leaves, so we cannot assign payoffs to leaves, as we normally do. Moreover, if the player were to get a small positive payoff with two strategies in each stage, then *both* of them will result in infinite payoffs with infinite stages, so that they are equivalent. This doesn't make intuitive sense! To get around the problem, with infinitely repeated games, we represent the payoff of a (potentially infinitely long) strategy as the average payoff per stage, assuming the limit exists. Alternatively, we can *discount* future expected payoffs at each stage by a factor $b$ that lies in [0,1]. That is, the payoff at the $i$th stage is multiplied by $b^i$. This often results in a payoff that is finitely bounded.

**Example 12: (Repeated game)**

Consider the two-stage version of the WiFi game in Example 6. With one stage, there are only two strategies for each player; that is, S and D, and four strategy profiles SS, SD, DS, and DD. With two-stages, they have four strategies each (SS, SD, DS, and DD) and therefore the normal form payoff matrix has 16 entries. The number of strategies for each player grows as $s^i$, where $s$ is the number of strategies available to each player, and $i$ is the number of stages. The payoff matrix has $s^{2i}$ entries. Thus, repeated games are cumbersome to represent even with a few stages.

The infinitely-repeated WiFi game can be similarly defined, with each player's strategy being an infinite string chosen from the alphabet {S, D}. Assume that the discount factor is 0.8. Then, if the expected payoffs for the row player are $r_0, r_1, r_2,...$ then the discounted net payoff for that player is $\sum_{i=0}^{\infty} r_i 0.8^i$ .

[]

There are many possible strategies for infinite-repeated games. A simple one is to always play the same pure strategy. A more interesting strategy is *tit-for-tat*, where each player plays what the other player played in the previous stage.

## 7.2 Solving a game

### 7.2.1 Solution concept and equilibrium

Intuitively, the *solution* of a game is the set of strategies we expect rational players to adopt, given the payoff matrix. The solution of a game is also called the *equilibrium*. The solution of a game *need not* be the strategy profile whose payoff to each player is greater than the payoff from any other strategy profile: as we will see shortly, rational players may sometimes be forced to play a strategy that actually gives them less payoff than some other strategy does! In common parlance, this is a ‘lose-lose’ situation.

The *solution concept* is the line of reasoning adopted in determining a solution or equilibrium. For the same game, different solution concepts can yield different equilibria. Game theorists usually make the simplifying assumption that all the players implicitly agree on using the same solution concept. If different players use different solution concepts the outcome is unpredictable.

**Example 13: (Solution of a game)**

It is easy to solve a game when there is a dominant strategy for one of the players. Consider the capture effect WiFi game of Example 10. We saw that the dominant strategy for the column player was S. This is known both to the column player and to the row player. Therefore, the row player can assume that, as a rational player, the column player will play S. Given this, the row player’s best response is D. Therefore, the solution to the game is DS, i.e., row player plays D and the column player plays S.

What about the game of Matching Pennies (Example 3)? We stated without proof that the best strategy for both players was a mixed strategy that equally randomized between H and T. This is also the solution of the game. Intuitively, this makes sense: if either player favoured H or T, then the other player would have a way to win more often than not. The only way to counteract this is to play randomly, with an equal chance of playing H or T. As an interesting aside, the Matching Pennies game also models penalty kicks in a soccer game. If the kicker and the goalkeeper both go left (L) or right (R), then the goalkeeper wins, otherwise the kicker wins. The best strategy for both, therefore, is to choose L or R with equal probability!

[]

### 7.2.2 Dominant strategy equilibria

The dominant strategy solution concept is used in games where each player has a strongly or weakly dominant strategy. The idea is simple: if each player has a strategy it should play irrespective of actions by other players, then it is reasonable to assume that the players will play this strategy. Note that the dominant strategy for a player may be a mixed strategy, i.e., a random variable defined over the domain of the pure strategies. When such an equilibrium exists, it is the preferred solution concept, because it makes the fewest demands on assumptions of player rationality.

**Example 14: (Dominant strategy equilibrium)**

Consider the WiFi game with delay-sensitive stations, for whom the cost of waiting one slot (-2) is worse than the cost of a collision (-1).This game is represented below in the normal form:

| | **S** | **D** |
|---|---|---|
| **S** | (-1,-1) | (1,-2) |
| **D** | (-2,1) | (-2,-2) |

TABLE 4. **The payoff matrix for the WiFi game with delay-sensitive stations**

Note that for the row player, strategy S always returns higher payoffs than strategy D, no matter whether the column player plays S or D. So, S is a dominant strategy for this player. Symmetrically, S is also the dominant strategy for the column player. Therefore, the dominant strategy equilibrium (solution) for this game is SS. It is interesting to note that a slight change in the payoff matrix completely the changes the outcome of the game!

[]

**Example 15: (Dominant strategy equilibrium: Prisoner’s Dilemma)**

In the game known as the ‘Prisoner’s Dilemma,’ two prisoners in two isolated cells are offered a bargain by the warden. If they inform on the other prisoner (defect or D), they are set free and the other prisoner is given four more years of prison. If neither informs on the other (showing solidarity or S), they only get a year of prison each. If both defect, they both get three years of prison each. What should each prisoner do?

We solve this game using the dominant strategy concept. The the payoff matrix as shown below:

|  | **S** | **D** |
|---|---|---|
| **S** | (-1,-1) | (-4,0) |
| **D** | (0,-4) | (-3,-3) |

TABLE 5. **The payoff matrix for the Prisoner's Dilemma game**

Note that the dominant strategy for the row player is to play D (because 0>-1 and -3>-4). By symmetry, D is also the dominant strategy for the column player. Therefore, the dominant strategy equilibrium is DD.

This game is called a 'dilemma' because the best possible outcome is SS. Nevertheless, the inexorable logic of game theory dictates that both prisoners will defect!

[]

### 7.2.3 Iterated removal of dominated strategies

In some games, not all players have dominant strategies, so it is not possible to find a dominant-strategy equilibrium. However, even if only one player has one or more dominant strategies (strategies whose payoffs are the same, and whose payoffs are greater than the player's other strategies), it may be possible to find a plausible equilibrium by deletion of *dominated* strategies. Specifically, if player *i* has a set of dominant strategies say $\{s_i^*\}$, then the other players can reason that player *i* will certainly play one of these strategies. Therefore, all of their own strategies incompatible with this set can be eliminated, which may then yield a dominant strategy set for one of the other players, say *j*. This, in turn, allows us to remove the dominated strategies for *j* and so on. In the end, we are hopefully left with a single strategy for each player, which is the equilibrium.

**Example 16: (Iterated removal)**

Consider the WiFi game with capture (Table 3 on page 146). Recall that here the column player has a dominant strategy S. The Row player can reason as follows: "Column will certainly play S. If Column plays S and I play S, I get -1 and if I play D, I get 0, so I should play D." Thus, with this solution concept, we can find the equilibrium DS.

[]

As with dominant strategies, not all games are guaranteed to have a dominant strategy equilibrium even with iterated equilibrium. For example, Matching Pennies does not have such an equilibrium. Also, applying this concept is tricky, because a pure strategy may be dominated by a mixture of other pure strategies, although none of the strategies in the mixture dominate the pure strategy (but none of the strategies in the mixture can be dominated by the pure strategy, either (Why?)).

### 7.2.4 Maximin equilibrium

The maximin equilibrium and its dual, the minimax equilibrium, are amongst the earliest and best-studied solution concepts. They both arise from the concept of *security level*, which is the guaranteed payoff even if the other players try their best to hurt you. The idea is to choose a strategy that maximizes this guaranteed payoff, thus maximizing the security level. It is, perhaps, a pessimistic, or even a paranoid way to play a game, in the sense that it does assume the opponents are rational: no matter what the other players do, the maximin payoff is guaranteed.

Mathematically, define $s_{-i}^{min}(s_i)$ to be the tuple of other player strategies that minimize the payoff to *i* when playing $s_i$. Then, the maximin strategy for *i* is $s_i^*$ where $s_{-i}^{min}(s_i^*) \geq s_{-i}^{min}(s_i')$ for all $s_i' \neq s_i^*$. Note that the maximin strategy for a player can be pure or mixed.

**Example 17: (Maximin strategy)**

Consider the WiFi game of Example 6. If Row plays S, then the Column player can play S to give it a payoff of -1, but if Row plays D, the worst payoff is only 0. So, the maximin strategy for Row is D. Similarly, the maximin strategy for Column is also D. Therefore, the maximin equilibrium for this game is DD. Note that there is no dominant strategy equilibrium for this game.

[]

**Example 18: (Maximin strategy graphically illustrated)**

Consider the following payoff matrix for a two-player zero-sum game:

| | C1 | C2 |
|---|---|---|
| R1 | (3,-3) | (1,-1) |
| R2 | (2,-2) | (4,-4) |

TABLE 6. **A zero-sum game**

We would like to compute the maximin strategy for both players. In general, this strategy is a mixed strategy, so assume that Row plays R1with probability *p* and R2 with probability (1-*p*). If Column plays C1, then Row gets a payoff of 3*p* + 2 (1-*p*) = *p*+2. If Column plays C2, then Row gets *p* + 4(1-*p*) = 4-3*p*. This is shown graphically in Figure 18. It can be shown that if Row uses any value of *p* other than 0.5, then it may obtain a payoff lower than 2.5 (see Exercise 14). It can also be shown that even if the column player plays a mixed strategy, Row can guarantee itself a payoff of 2.5 by playing 0.5R1+0.5R2. Similarly, it can also be shown that the column player can hold the row player down to 2.5 by playing 0.75C1+0.25C2. If the column player plays any other strategy, the row player can obtain a greater payoff (and, because this is a zero sum game, the column player will get a lower payoff). Therefore, the maximin strategy for Column is 0.75C1+0.25C2 and the mixed strategy profile (0.5R1+0.5R2, 0.75C1+0.25C2) is the maximin equilibrium for this game.

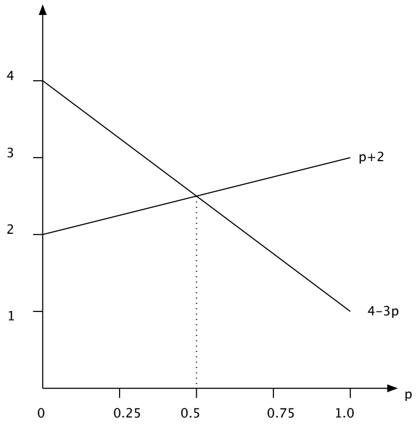


FIGURE 18. **Maximin strategy for the Row player (Example 18)**

[]

In a two-person game, the equilibrium point is also called a *saddle* point. Any deviation from this point by the Row or Column player will decrease its guaranteed minimum payoff (though they move away from this point in orthogonal dimensions).

The dual of the maximin strategy is the minimax strategy. In the two-player version, a player acts so as to minimize the best possible payoff that the other player receives. In this sense, the player acts to maximally punish the other player without regard to their own payoff. The *n*-player version of this solution concept is somewhat more tricky to state and requires the players to act as if they form a coalition to punish a given player. This solution concept is therefore of only theoretical interest, in that it is the dual of the maximin strategy, so we will not consider it any further.

One of the earliest theorems in game theory was stated and proved by von Neumann in 1928 and is called the Minimax theorem. It states that in every finite, two-person, zero-sum game, player 1 is guaranteed a payoff of at least *v* independent of player 2's strategy, and, symmetrically, player 2 can restrict player 1 to a value of at most *v*, independent of player 1's strategy. This value may require players to play mixed strategies. The strategy corresponding to this guaranteed payoff is the maximin strategy. Because player 1 is guaranteed to get at least *v*, and player 2, being an opponent in a zero-sum game, and thus receiving a payoff of -*v*, would never want player 1 to get any payoff higher than *v*, this is an equilibrium. Thus, another way to view this theorem is that it asserts that every finite, two-person, zero-sum game has a equilibrium that results from both players playing the maximin strategy.

### 7.2.5 Nash equilibria

Although dominant strategy equilibria are preferred, many games do not allow such a solution. For instance, there is no dominant strategy solution for the Matching Pennies game or the WiFi game of Example 6. In such games, we turn to a different solution concept, called the *Nash* equilibrium concept.

The key idea in a Nash equilibrium is that players have no incentive to deviate from it. That is, *assuming every other player* is playing the strategy corresponding to the Nash equilibrium strategy profile, no player's payoffs are better by choosing any other strategy. Mathematically, a strategy profile $(s^*_1, s^*_2,...,s^*_n)$ is a (weak[1]) Nash equilibrium if:

$$\pi_i(s^*_i, s_{-i}) \geq \pi_i(s_i, s_{-i}) \qquad \forall s_i \neq s^*_i \qquad \textbf{(EQ 63)}$$

In a Nash equilibrium, each player plays its best response assuming all other players play their Nash strategy. This exposes both the power and weakness of the concept. The concept is powerful because it intuitively matches our expectations of a rational player who plays the best response to the other players' actions. Moreover, we only need to identify a potential Nash strategy profile (it doesn't matter how) and check whether any player has an incentive to deviate, which is straightforward. However, a Nash equilibrium pre-supposes that all players are going to play according to this solution concept. Worse, it is not unique: a game may have more than one Nash equilibrium. In this case, players need to either guess which Nash equilibrium will be chosen by the others, or coordinate their actions using an external signal.

Every dominant strategy equilibrium, by definition, is also a Nash equilibrium (though the converse is not true) (Why?). For instance, in Prisoner's Dilemma, DD is both a dominant strategy and a Nash equilibrium. SS is not a Nash equilibrium because if a player assumes that the other player is going to play S, it pays for it to defect. Similarly, every maximin equilibrium is also a Nash equilibrium (Why?).

Note that a Nash equilibrium may involve mixed strategies, as the next example shows.

**Example 19: (Nash equilibrium for Matching Pennies)**

Recall the Matching Pennies game (Table 1 on page 143). We will prove that the Nash equilibrium is for both players to play a mixed strategy with the probability of H (or T) = 0.5 (represented as 0.5H + 0.5T).

Consider the situation for the row player. Because we are solving for a Nash equilibrium, we can assume that the column player plays 0.5H + 0.5T. Let the row player play *p*H + (1-*p*)T. Row plays H with probability *p*. We have two cases: Column plays H or Column plays T. (a) Column plays H with probability 0.5, giving Row a payoff of 1, so that the expected payoff for Row in this case is *p*. (b) Column plays T with probability 0.5, giving Row a payoff of -1, so that the expected payoff for Row in this case is -*p*. Thus, the expected payoff when Row plays H is 0. Arguing along the same lines, the expected payoff for Row when it plays T is also 0. So, we have the interesting situation that, if Column plays 0.5H + 0.5 T, then no matter what Row does, its expected payoff is 0. By symmetry, if Row plays 0.5H+0.5T, then the expected utility of Column is 0, no matter what it does. Therefore, Equation 63 holds (albeit with equality, rather than inequality) and this is therefore a (weak) Nash equilibrium.

[]

**Example 20: (Finding Nash equilibria: Battle of the Sexes)**

[1] The equilibrium is strict if the inequality is strict.

Consider the following *coordination* game, popularly called the 'Battle of the Sexes.' A couple want to choose between going to a prizefight (which the row player wants) and going to the ballet (which the column player wants)[2]. Both would rather be with each other than by themselves.

| | F | B |
|---|---|---|
| **F** | (2,1) | (0,0) |
| **B** | (0,0) | (1,2) |

TABLE 7. **The payoff matrix for the 'Battle of the Sexes' coordination game**

It is obvious that the two pure-strategy Nash equilibria are FF and BB. There is, however, also a mixed-strategy Nash equilibrium where Row plays 2/3F + 1/3 B and Column plays 1/3F + 2/3B. To see this, assume that Column plays 1/3F + 2/3B and Row plays $p$F + (1-$p$)B. Then, Row's expected payoff is $p$(1/3*2) + (1-$p$)(2/3*1) = 2$p$/3 + 2/3 -2$p$/3 = 2/3, independent of $p$. Symmetrically, it can be shown that when Row plays 2/3F + 1/3B, Column always gets 2/3 independent of its strategy. Therefore, Equation 63 holds with equality, and this is a weak Nash equilibrium. It is worth noting that this mixed equilibrium gives a lower payoff than either of the pure equilibria.

[]

In 1951, J. Nash proved a famous theorem that earned him the Nobel prize in economics. It states that every game with a finite number of players and actions has at least one Nash equilibrium. Thus, we can always find at least one Nash equilibrium for finite normal-form games. The Nash theorem, however, only proves existence of an equilibrium (by relying on a fixed point property of iterated maps described by the Brouwer fixed-point theorem) rather than telling us how to find this equilibrium. Therefore, in practical cases, finding the actual equilibrium can be challenging. In doing so, however, it is useful to rely on the following fact: if a mixed strategy uses strategy $s$, then $s$ cannot be strongly dominated by any other strategy (otherwise, you could remove this strategy from the mixture and increase the payoff).

So far, we have considered Nash equilibria in games with discrete actions. However, a player's actions may be chosen from a subset of the real line. Finding Nash equilibria in such games is essentially the same as finding local maxima of a vector-valued function of a vector of real variables. More precisely, consider the mapping $G$ from a vector $\mathbf{x}$ in $\mathbf{R}^n$, corresponding to the action chosen by each player, to the payoff vector $\pi$ which determines the corresponding payoff to each player. We say that a payoff vector $\pi_i$ dominates payoff vector $\pi_j$ if every element of $\pi_i$ is greater than or equal to the corresponding members of $\pi_j$. At each local maximum $\mathbf{x}_m$ of $G$, $\pi_m = G(\mathbf{x}_m)$ dominates $\pi_j = G(\mathbf{x}_j)$ for all $\mathbf{x}_j$ in the neighbourhood of $\mathbf{x}_m$. We can use any standard optimization technique, even a heuristic such as hill climbing, to find these local maxima. If $G$ is globally convex, that is, there is a single global maximum, then *any* hill climbing technique will find the global maximum. In many papers on game theory, $G$ is assumed to be convex, so that this approach can be used. It is an open question, however, whether this convexity assumption holds in the real world.

### 7.2.6 Correlated equilibria

A fundamental problem with the Nash equilibrium concept arises when a game has more than one Nash equilibrium, when each player has to somehow guess which of the game's equilibria the others pick. Even if there is a unique equilibrium, each player has to know exactly which strategy every other player is going to play. If the other players are playing mixed strategies, each player also has to know exactly how every other player is going to mix each of their pure strategies. This isn't very realistic!

A hint to a possible solution lies in the formulation of a Bayesian game, where a move by Nature decides player types and each player has the same subjective probability distribution over the types chosen by Nature. (Once the player types are chosen, however, the rest of the game is a standard game and is solved using a standard solution concept.) The key idea of a correlated equilibrium is to extend the subjective probability distribution of each player not just to player types but also to player *strategies,* given a *shared* view on the current state of the world. That is, each player has a subjective probability distribution

[2] This being the twenty-first century, we'll leave the sex of the row and column players anonymous.

over possible strategies of the other players, conditional on a random variable called the 'state of the world' and tries to maximize their own payoff conditional on this distribution. In the original formulation of this solution concept, the shared common view on the state of the world was expressed in the form of an external, globally-trusted, correlating agent who tells each player what to do. Such an agent is not really necessary, as shown in [R. Aumann, Correlated Equilibrium as an Expression of Bayesian Rationality. Econometrica 55(1):1-18, 1987]. Nevertheless, it is easier to discuss a correlated equilibrium assuming such an agent, and we shall do so as well.

More precisely, we assume the existence of an external agent that tells each player which pure strategy to play. It does not tell the player, however, what it told the other agents (this models the fact that each player has their own subjective probability distribution on their pure strategies as a function of the state of the world). The player then plays this strategy. We say that the resulting strategy profile is a *correlated equilibrium* if the players do not have any incentive to deviate from this strategy.

**Example 21: (Simple correlated equilibria)**

Recall from Example 15 that in the Prisoner's Dilemma (Table 5 on page 149) both the Dominant Strategy and Nash equilibrium is DD, with payoff (-3,-3). Suppose we introduce correlation through a coin toss. If the coin lands heads up, then the players are told to play DS, and if tails up, they are told to play SD. Note that when told to play S, Row would gain by playing D instead! So, it has an incentive to deviate, and this is *not* a correlated equilibrium.

Now, consider the Battle of the Sexes game (Table 7 on page 152). Suppose an outside agent told each player to play FF or BB based on the results of a coin toss. Consider the Row player's point of view. If it is told to play F, then it does pay for it to deviate and play B. Symmetrically, neither does Column gain from deviation. Therefore, this is a correlated equilibrium. If the coin is fair, both can achieve an expected gain of 0.5*2 + 0.5*1 = 1.5, which is more than they can gain from a mixed Nash strategy, which gives them only 2/3 each.

[]

Correlated equilibria can be complex if the correlating device does not allow a player to determine exactly what the other players would do, as was the case with the Battle of the Sexes. This is illustrated by the following game.

**Example 22: (Correlated equilibrium for 'Chicken')**

The game below models the game of 'chicken' where two racers rush towards each other at full speed. The first person to pull away is the chicken and loses. Of course, if neither pulls away, both lose. The game matrix is given below, where D stands for 'dare' and C for 'chicken.'

| | **D** | **C** |
|---|---|---|
| **D** | (0,0) | (7,2) |
| **C** | (2,7) | (6,6) |

TABLE 8. **Payoff matrix for the Chicken game**

We assume the existence of an external agent, and we assume that both players know that the external agent says CC with probability 1/3, DC with probability 1/3, and CD with probability 1/3 (and never says DD). Suppose Row is told to play D. Then, it knows that Column was asked to play C. Row goes from 7 to 6 by deviating, so it will not deviate. Suppose Row is told to play C. Then, there is probability 1/2 that the other player was told to play D and probability 1/2 the other player was told to play C. Assuming the other player does not deviate from the correlated equilibrium, the expected utility of deviating and playing D is 0(1/2) + 7(1/2) = 3.5 and the expected utility of listening to the agent and playing C is 2(1/2) + 6(1/2) = 4. So, the player would prefer to play C, i.e. not deviate. From the symmetry of the game, the same argument holds for the Column player. Therefore, neither player will deviate from the suggestion of the agent, and this is a correlated equilibrium.

[]

Note that in a correlated equilibrium, we have a free variable, which is the probability with which the advisor asks each player to play each pure strategy (from which none of the players will deviate). By choosing different values for this distribution, we can achieve a range of payoffs to each player.

The correlated equilibrium concept is more appealing than a Nash equilibrium, in that it does not require players to know the exact strategy for every player, just that they will condition on the same 'state of the world.' Moreover, every Nash equilibrium (whether using pure or mixed strategies) can be shown to be a correlated equilibrium that only advises pure strategies. Therefore, correlated equilibria are the more powerful solution concept, especially if an external correlating agent can be naturally found in the problem domain.

### 7.2.7 Other solution concepts

Our treatment of solution concepts is necessarily limited. Several concepts such as rationizability, sub-game perfectness, trembling-hand perfectness, ε-Nash equilibria, and evolutionary stability have been studied in the literature. More detail on these can be found in a standard text such as [Martin J. Osborne and Ariel Rubinstein, A course in game theory (MIT Press, 1994)].

## *7.3 Mechanism design*

Traditionally, game theory studies the behavior of players when the game has already been specified. In contrast, *mechanism design* sets up the rules of a game such that rational (utility maximizing) players, in equilibrium, will behave as the designers intended. The key idea is to choose the rules of the game so that each player's attempt to maximize their utility also achieves the desired outcome *without* the designer knowing each player's utility function. Colloquially, we 'want the players to do what *we* want them to do because *they* want to do it!'

### 7.3.1 Examples of practical mechanisms

Mechanism design arises naturally in some common situations. Consider the owner of a good who wants to sell it to the buyer who will pay the most for it. This can be achieved by an auction mechanism that treats the bidders as players in a game. In equilibrium, the mechanism ensures that a player's bid reflects their true valuation of the good. If they value it highly, it will be in the player's own best interest to bid high instead of untruthfully bidding lower than their true value in an attempt to pay a lower price.

As another example, consider an election officer who wants to choose one of the candidates standing for election as the winner. Presumably the winner should, in some way, reflect the wishes of 'society.' The electoral officer implements what is known, in the literature, as a *social choice function*. Again, we desire a mechanism such that, assuming that such a social choice exists in the first place, each voter reveals their true preferences (instead of voting 'strategically,' that is, voting other than for their choice in an attempt to influence the final outcome).

It turns out that similar considerations arise in several networking problems. For instance, if several users want to download a file using BitTorrent, how can we ensure that every user does their fair share in downloading and sharing the torrent? Similarly, if we have a group of mobile phone users sharing content using ad hoc phone-to-phone data propagation, how can we make sure that every user has an incentive to participate in the scheme despite using their scarce battery resources? The general area of mechanism design addresses these issues. Due to considerations of space, we will merely touch upon the main concepts. For an excellent discussion that goes deeper into these concepts, please refer to [Parkes thesis Chapter 2; Algorithmic Game Theory, Chapter 9, Cambridge University Press, 2007].

### 7.3.2 Three negative results

In using mechanism design to achieve social choices, it is useful to keep three negative results in mind: the Condorcet Paradox, Arrow's theorem, and the Gibbard-Satterthwaite theorem

The *Condorcet Paradox* shows that even in an election with just three voters and three candidates, we cannot find a self-consistent majority or 'social' choice. Consider an election with three candidates *a*, *b*, and *c*. Let voter 1 prefer *a* to *b* and *b* to *c*. Let voter 2 prefer *b* to *c* and *c* to *a*. Finally, let voter 3 prefer *c* to *a*, and *a* to *b*. Now, note that two voters (1 and 3) prefer *a* to *b*. So, the majority prefer *a* to *b*, and in the social choice, it must certainly be the case the *a* is preferred to *b*. However, the voter preferences are chosen to be rotationally symmetric, so a majority also prefers *b* to *c* and *c* to *a*. If preferences are transitive, which is certainly necessary for consistency, then we find that a majority prefers *a* to *b* to *c* to *a*! Thus, in reflecting a society's choice of candidates, we have to give up either majority (i.e. have a 'dictatorship'), or consistency, or transitivity. None of these are appealing choices. One may think that the problem here is that we are using a simple majority rule. What if this were replaced with something more sophisticated, such as proportional representation? Indeed, several schemes, called *voting methods*, have been proposed in the literature that are more sophisticated than majority and remedy some of its problems. Although they each have their strengths, they all run afoul of a fundamental theorem of social choice, called Arrow's theorem.

*Arrow's theorem* states that under some very general conditions, we cannot reasonably compose individual *strict* orderings of alternatives (i.e., orderings where every alternative has a definite rank, though multiple alternatives may have the same rank) to form a global strict ordering on alternatives. The theorem assumes that individual preferences are arbitrary: one individual may rank alternative 1, for example, as its most-preferred alternative, but another may rank the same alternative as its least-preferred alternative. Define a *social welfare function* as a function that aggregates individual strict orderings of a finite set of alternatives to form a global ordering. We say that a social welfare function satisfies *unanimity* if it orders alternative *a* higher than *b* iff every individual ranks *a* higher than *b*. A social welfare function satisfies *independence of irrelevant alternatives* if the social (aggregate) ranking of two alternatives *a* and *b* depend only on how each individual ranks *a* and *b*. That is, the other alternatives could be arbitrarily ranked, but as long as every individual ranks *a* higher than *b*, so should the social welfare function. Finally, we call a social welfare function a *dictatorship* if there is an individual *i* such that the social welfare function's choice of the top-ranked alternative is that individual's choice of the top-ranked alternative, no matter what the other individuals desire[3]. Arrow's theorem states that every social welfare function over a set of more than two alternatives that satisfies unanimity and independence of irrelevant choices is a dictatorship! This is troublesome, in that we have to give up either unanimity or independence of irrelevant choices to avoid dictatorship.

The third minefield in the design of social choice and social welfare functions is called the *Gibbard-Satterthwaite theorem* (which can also be formulated as an extension of Arrow's theorem). Returning to social choice functions, that is, functions that decide on a choice of an alternative (rather than an ordering on alternatives), we call a social choice function *strategically manipulable* by individual *i* if that individual can influence the outcome of the social choice function in their favour by misrepresenting their preferences (i.e., lying about their preference ordering). However, if a social choice function cannot be manipulated by an individual in this manner, it is called *incentive compatible*. We will now show that if we have an election with two candidates, standard majority voting is incentive compatible. First, note that a voter can manipulate the outcome if and only if there are an odd number of voters and this voter is casting the deciding vote. If the voter lies when they cast the deciding vote, they do not influence the election in their favour, indeed, the outcome is the opposite of what they want. Therefore, the outcome is non-manipulable, that is, incentive compatible. The Gibbard-Satterthwaite theorem states that if *f* is an incentive-compatible social choice function that decides between more then two alternatives, then *f* is a dictatorship, that is, there is some individual who can control the outcome of the election. The consequence of this theorem is that when aggregating individual choices, expressed in the form of total orderings, no matter how clever our scheme, we are going to either allow it to be manipulated, or allowed it to be dictated to! This is a strong negative result.

There are several ways out of this quandary. One of them, which turns out the basis of most approaches to mechanism design, is to introduce the notion of money. Specifically, in Arrow's framework, the utility of each individual is expressed only through its preferred ordering, and, in fact, such a simplistic notion of utility is necessary for the Gibbard-Satterthwaite theorem. If, however, we assume that an individual's utility is *quasi-linear*, where the utility depends not only on the alternative selected, but also on an additional monetary side payment, then preferences for alternatives cannot be arbitrary and both Arrow's and the Gibbard-Satterthwaite theorem can be avoided. In the context of voting, what this means is that an individual whose choice did not affect the final outcome would be compensated by a certain amount of money. So, the greater the attempt by a voter to change other voters' choices, the more it will cost. Therefore, assuming that everyone has the same amount of money, we can avoid manipulations (or at least, buying elections will only be for the rich!).

[3] This is a weak form of dictatorship, in that if the other individuals were to change their preference orderings, the identity of the dictator could change. The idea is that any voting method that meets Arrow's criteria necessarily transfers the power to decide the social choice, i.e, cast the 'deciding vote,' one of the (perhaps unwitting) individuals.

### 7.3.3 Two examples

To fix ideas, we will first study two simple examples of mechanism design, where a *principal* designs a game so that the players or *agents* do the right thing.

**Example 23: (Price discrimination)**

Suppose you manufacture a communication device called the uPhone that can be bought by one of two types of customers. Chatters (C) need only one uPhone, because most of their friends already have one, and Antediluvians (A) need at least two, one for each party making a call. What price should you set for them?

A naive solution would be to price the uPhone at, say, \$100, so that C pays \$100 and A pays \$200. Suppose the internal value that C ascribes to a uPhone is \$50, and the internal value that A ascribes to two uPhones is \$300. By pricing it at \$100, no C will buy it, and every A who buys it would have been willing to pay \$150 per uPhone, so that you are leaving money on the table. Can you do better?

If you knew that the internal valuation of C for a uPhone was $c$ and of A was $a$ and if $2a > c$, then you could price the uPhones as follows:

If you buy one uPhone, it costs $c$ but if you buy two it costs min($2a$, 2c)

This way, chatters would pay $c$, so you would not lose sales to them. Since $2a > c$, they are never tempted to get two when they don't need it. If $2a > 2c$, and you price two uPhones at greater than $2c$, then Antediluvians would just buy two uPhones individually, so there is no point in setting the price for two any higher than $2c$. On the other hand, if the price for two is more than $2a$, then no As will buy uPhones. So, the price for two should be the smaller of *2a* and *2c*. This *discriminative pricing scheme* gives a seller the most possible profit and the largest possible customer base.

[]

Note that for this scheme to work, we need to know the internal valuation of each type of customer. But this is private information: how can a seller determine it? A hint to a solution can be found in the *Vickrey* or *second price* auction.

**Example 24 (Vickrey auction)**

Consider the sale of a good by auction. Unlike the previous example, assume that the seller does not know the internal valuations of the buyers (in which case the solution is trivial). One possible solution--a Vickrey auction--is to ask buyers to bid, award the good to the highest bidder, but charge the winner the second-highest bid. We now prove that this results in each buyer telling the seller its true internal valuation.[4]

If a bidder is going to tell a lie, they can either (A) bid a higher valuation or a (B) bid a lower valuation than their true valuation. Suppose they bid a higher valuation. Now we have two more cases: either (A.1) they win the auction or (A.2) they lose. If they win, their utility depends on the second price:

- (A.1.a) If the second price was below their own valuation, they gain a utility corresponding to the difference between their true valuation and the second price. But they would have obtained the same gain by telling the truth, so telling a lie doesn't help.
- (A.1.b) If the second price was higher than their own true valuation, they lose utility.

If they lose (A.2), of course, telling a lie does not help, since the utility from the transaction is zero[5].

Now, suppose they bid a lower valuation (case B). Again, we have two sub-cases: either they win (B.1) or they lose (B.2).

[4] It does not, unfortunately, result in the seller getting the best price. If the second-highest bid is very low, the seller may end up with essentially nothing, although there was a willing buyer at a higher price. This is why Vickrey auctions are rarely used in practice.

[5] Of course, a player who knows the internal valuation of other players could artificially boost up their price to just below that of the (known) winning bid, to hurt the winner. But, this violates the assumption that internal valuations are secret.

- (B.1) If they win, they pay the second price, and by reasoning along the same lines as cases A.1.a and A.1.b, we can see that telling a lie either hurts or is as good as telling the truth.
- (B.2) If they lose, the lie actually hurt them in that they could have won, but didn't.

We have shown, therefore, that in all cases, telling a lie is either as good as or worse than telling the truth. Therefore, a rational player would tell the truth and reveal its true internal valuation to the seller. This is called *truth revelation*.

Note that the price obtained by the seller is not as high as in the previous example (where the seller would have obtained the internal valuation of the highest bidder). Nevertheless, this simple scheme has the remarkable property that the design of the game (mechanism) makes it incentive compatible for a rational player to reveal the truth. This is at the heart of all mechanism design.

Importantly, we require the players to care about how much money they bid, that is, their utilities are quasi-linear. If this was not the case, due to the Gibbard-Satterthwaite theorem, we would end up with a dictatorship (where only player would decide who gets the good) or would have the price of the good strategically manipulable by one or more players.

[]

### 7.3.4 Formalization

We now formalize the intuitions we have developed so far. We assume that there is a principal $P$ who is designing a mechanism with $n$ agents, indexed by $i$. We assume that the mechanism is associated with a set $O$ of one of $|O|$ outcomes (chosen by the principal), with each action called $o$. For instance, in the uPhone example, at any given time, there is one agent playing the game (the customer in the store), so $n$=1. The set of possible outcomes $O$ is $\{(0,.), (1,c), (2,\min(2a, 2c))\}$, where the first tuple represents "don't purchase," the second outcome represents "purchase 1 for $c$" and the third outcome represents "purchase 2 for min($2a$, $2c$)." We assume that each agent has a type $t_i$ which captures all the private information relevant to their decision-making. For instance, in the uPhone example, the type was A or C. Each $t_i$ is drawn from a set of all possible types for the $i$th player, $T_i$. We assume that each agent has a preference ordering over the outcomes, which are represented in the form of private, quasi-linear utility functions $U_i(o, t_i)$. For example, the utility function of a chatter is $U_c((1,c), c) = c\text{-}c = 0$; $U_c((2, \min(2a, 2c)), c) = \min(2a, 2c) - c < 0$. (Why?).

We define the possible 'states of the world' as the product of all possible agent types, and denote it $T = T_1 \times T_2 \times ... \times T_n$, where '$\times$' is the cross product. A s*ocial choice function* is a mapping $f$ from $T$ to $O$, that describes, for each possible state of the world, the desired outcome. This is outcome that the principal would have chosen *assuming that it knew the true state of the world*. In the uPhone example, there is a single agent (customer) at any given time, who has type A or C. The social choice function $f$ maps C to $(1,c)$ and A to $(2,\min(2c, 2a))$. Of course, a principal does not know the true types. So, we seek a mechanism that results in the right outcome (i.e., *implements f*) without knowledge of the state of the world.

A *mechanism* is an $n$-tuple of strategy spaces (also called message or action spaces) $S = S_1 \times S_2 \times ... \times S_n$ and an *outcome* function $g$ that maps from $S$ to $O$. Each $S_i$ represents the possible strategies (or actions) allowed to an agent in the mechanism, that is, the rules of the corresponding game. In the uPhone example, $S_A = S_D$ = {buy nothing, buy one uPhone, buy two uPhones}. The function $g$ represents the outcome as a function of the agent's strategies. In the uPhone example, this is the pricing schedule, that maps from 'buy one' to price $c$ and from 'buy two' to price min($2a$, $2c$). Recall that each player has utilities over these outcomes.

A mechanism $M = (S_1,...,S_n, g(.))$ is said to *implement* social choice function $f(T)$ if there is an equilibrium strategy profile $s^*=(s_1^*(t_1),...,s_n^*(t_n))$ of the game induced by $M$ such that:

$$g(s_1^*(t_1),...,s_n^*(t_n)) = f(t_1,...,t_n) \qquad \textbf{(EQ 64)}$$

The equilibrium of the game depends on the underlying solution concept. The most widely used concept is dominant strategy. However, in some cases, a Nash equilibrium (no agent will deviate from the equilibrium) or a Bayes-Nash equilibrium (the expected gain to each agent at the Nash equilibrium exceeds the expected utility from deviation) is also used. We will only study dominant strategy solutions here because they are more plausible than the other solution concepts. In this solution concept, letting $s_{-i}$ represent the strategies of players other than $i$:

$$u_i(g(s_i^*(t_i), s_{-i}^*(t_{-i})), t_i) \geq u_i(g(s_i^/(t_i), s_{-i}^/(t_{-i})), t_i) \qquad \forall i, \forall t_i, \forall s_i^/ \neq s_i^*, \forall s^/_{-i} \qquad \textbf{(EQ 65)}$$

This implies that for player *i,* no matter what the other players play, the utility from the dominant strategy is as much as or greater than any other strategy.

### 7.3.5 Desirable properties

We now define certain desirable properties of any mechanism. Note that these are not mutually compatible: we need to balance between them in any practical mechanism.

Individual rationality: No agent should be forced to participate in the mechanism: every agent should receive a greater utility from participation than non-participation. (For Bayesian agents, these would be expectations rather than actual utilities, conditioned on each agent's prior knowledge of (potentially) their own type, and the types of the other agents.)

Strategy-proofness: A mechanism is strategy-proof if it is incentive-compatible and the equilibrium is a dominant-strategy equilibrium.

Incentive compatibility: Roughly speaking, a mechanism is incentive compatible if it is in the best interests of each agent to cooperate. More precisely, if the designer of the mechanism would like agent *i* to play strategy $s_i^*$ in a dominant-strategy equilibrium, then the mechanism is incentive compatible if, in such an equilibrium, the payoff to agent *i* when it plays $s_i^*$ is as good as or better than the payoff with any other strategy.

Efficiency: A mechanism is efficient if the outcome that is selected maximizes the total utility. At first glance, this seems impossible: after all, a player's utility is private! However, recall our assumption that the principal knows the form of the utility function of each player and all it does not know is a type parameter. So, the operation is possible. A second objection is whether summing utilities is meaningful given that utilities are only defined up to an affine transform. Here, the introduction of a common 'money' parameter that all agents value (which we will soon see), allows us to plausibly maximize the sum of the utilities.

Budget-balance (BB): In general, a mechanism may require transfers of money between agents. A mechanism is budget-balanced if these net transfers (across agents) are zero, so that the principal does not have to inject money into the system. (When dealing with Bayesian agents, we need to distinguish between *ex ante* budget balance, which means that the budget is balanced only in expectation. With *ex post* budget balance, the budget is balanced every time.)

Fairness: In some cases, we would like the mechanism to select the outcome that minimizes the variance in the utilities of the agents. This is defined as 'fairness.'

Revenue maximization: Obviously, the designer of the mechanism would like to get the most possible revenue from the mechanism.

Pareto Optimality: A mechanism is Pareto Optimal if it implements outcomes where increasing the utility of any agent would necessarily decrease the utility of some other agent. That is, there is no 'slack' in the system. With quasi-linear utilities, this turns out to be the same as efficiency.

### 7.3.6 Revelation principle

In a general mechanism, it is possible for players to have arbitrarily complex strategy spaces. Consider the following particularly simple strategy called *direct revelation*: the agent tells the principal its type $t_i$. Of course, the agent could lie. Nevertheless, it should be clear that revelation greatly restricts the strategy spaces and simplifies the mechanism.

Formally, a direct-revelation mechanism $M = (T_1,...,T_n, g(.))$ is a mechanism where $S_i = T_i$, i.e., the strategy space for agent *i* is its set of valid types. A direct-revelation mechanism is incentive compatible if, in equilibrium, the chosen strategy is to tell the truth, i.e., $s_i(t_i) = t_i$ for all $t_i$ in $T_i$. Note that in a direct-revelation mechanism, the outcome function *g* is the same as the social choice function *f*, because they both operate on the same space of agent types.

Now, suppose we restrict mechanisms to only those where the only strategy allowed to an agent is direct revelation. Do we give up anything? Are there mechanisms that are more complex and can therefore achieve outcomes that this simple mechanism cannot? The surprising answer is that there are not! Every mechanism, no matter how complex, that achieves its goals through a dominant strategy equilibrium can be reduced to a mechanism where the only strategy for an agent is direct revelation and the solution concept is dominant strategy equilibrium.

To see this, note that the complex mechanism must require the player to play *some* strategy $s_i^*(t_i)$ in equilibrium. Not only are the strategies that are allowed each agent under the control of the principal but also these strategies can only depend on $t_i$. Therefore, the principal could simulate $s_i$ if it were told $t_i$. Thus, no matter how complex $s_i$, all the agent needs to tell the principal is $t_i$ and the principal would compute the same outcome in equilibrium as would the complex mechanism. The preceding reasoning, with a modicum of mathematical formalism, is easily proved, and is known as the *Revelation Principle*.

Given this principle, we need only study mechanisms of the direct-revelation type. Moreover, we would like to design mechanisms where truth-telling is the dominant strategy, or, in short, direct-revelation incentive-compatible mechanisms. But, do any such mechanisms exist? The answer is affirmative, as the next section shows.

### 7.3.7 Vickrey-Clarke-Groves mechanism[6]

The Vickrey-Clarke-Groves (VCG) mechanism is a direct-revelation mechanism that makes truth-telling incentive compatible. Because all mechanisms that use dominant-strategy as a solution concept can be reduced to an equivalent direct-revelation mechanism, the VCG mechanism is a widely-used building block in the design of dominant strategy mechanisms.

In the VCG mechanism, each agent tells the principal its (purported) type. Based on these types, the principal computes an outcome $x$ (i.e., the social choice). It also asks each agent to make a payment $p_i$. The agent would not like to pay the principal any money, so its utility declines with increasing payments. By choosing payments carefully, the principal can make truth telling the dominant strategy equilibrium of the game, so that the social choice that is computed based on reported types is the true social choice.

Specifically, given an outcome $x$, the VCG mechanism assumes that agents have quasi-linear utility functions of the form

$$u_i(x, p_i, t_i) = v_i(x, t_i) - p_i \qquad \textbf{(EQ 66)}$$

where the principal knows the form of $v_i$ but not the parameter $t_i$. The $v$ function is called the *valuation* function, and it describes how highly each agent values the outcome ('public good') $x$, based on its type. Agent $i$ of true type $t_i$ tells the principal that its type is $\hat{t}_i$. The principal computes the social choice or outcome $x^*$ as:

$$x^* = g(\hat{t}_1, \ldots, \hat{t}_n) = \underset{x}{\text{arg max}} \sum_i v_i(x, \hat{t}_i) \qquad \textbf{(EQ 67)}$$

Thus, $x^*$ is chosen to maximize the sum of individual valuations as a function of the reported types. Note that this potentially makes $x^*$ strategically manipulable, in that an agent may report a type that would make $v_i(x^*, \hat{t}_i)$ be more in line with $i$'s wishes (making $i$ a dictator). To avoid this, the VCG mechanism asks each player to make a payment $p_i$, where

$$p_i = h_i(\hat{t}_{-i}) - \sum_{j \neq i} v_j(x^*, \hat{t}_j) \qquad \textbf{(EQ 68)}$$

where $h(.)$ is any function that is independent of $t_i$. The key idea is to pay agent $i$ an amount equal to the sum of the other player's valuations. So, given that the social choice is $x^*$, the agent $i$'s utility becomes:

[6] The Clarke, Groves, and Vickrey mechanisms differ slightly in their generality. For simplicity, we will refer to all three interchangeably as VCG mechanisms.

$$\begin{aligned} u_i(x^*, p_i, t_i) &= v_i(x^*, t_i) - p_i \\ &= v_i(x^*, t_i) - \left( h_i(\hat{t}_{-i}) - \sum_{j \neq i} v_j(x^*, \hat{t}_j) \right) \\ &= -h_i(\hat{t}_{-i}) + v_i(x^*, t_i) + \sum_{j \neq i} v_j(x^*, \hat{t}_j) \end{aligned} \qquad \textbf{(EQ 69)}$$

Of these three terms, agent *i* has no control over the first term, because *h* does not depend on *i*. Similarly, it has no control over the third term, which sums over the valuations of the other agents. It can only control the second term by its choice of reporting its type. It should therefore report a type that maximizes the value of $v_i(x^*, t_i)$. How can it do that? Recall that the mechanism finds *x* as the value that maximizes $\sum_i v_i(x, \hat{t}_i) = v_i(x^*, \hat{t}_i) + \sum_{j \neq i} v_j(x^*, \hat{t}_j)$ . Comparing this with Equation 69, we see that agent *i* can maximize its utility by making $v_i(x^*, t_i)$ the same as $v_i(x^*, \hat{t}_i)$, and this will happen only if $\hat{t}_i = t_i$, that is, it tells the truth.

Essentially, the VCG mechanism forces each agent's utility function to be the sum of the reported valuations of all the users. Thus, every agent reports its type truthfully so that the overall maximization is in its own favour. This makes truth-telling incentive compatible, so VCG is not strategically manipulable. Moreover, this is the only known mechanism that provides both individual rationality and efficiency.

We have thus far left *h*(.) undefined. Different choices of *h*(.) can achieve different outcomes. For example, it can be used to achieve *weak budget balance* (the principal may net revenue, but will never have to pay money out), or individual rationality (no agent will be worse off participating than not participating). There is a particularly well-chosen value of *h*, called the *Clarke Pivot* value that guarantees individual rationality while also maximizing revenue for the principal (but not necessarily budget balance), that we describe next.

First, define $x^{-i}$ as the social choice computed without taking agent *i*'s input into account:

$$x^{-i} = \underset{x}{\arg\max} \sum_{j \neq i} v_j(x, \hat{t}_j) \qquad \textbf{(EQ 70)}$$

Then, the Clarke Pivot price that *i* pays, $p_i$ is given by

$$p_i = \sum_{j \neq i} v_j(x^{-i}, \hat{t}_j) - \sum_{j \neq i} v_j(x^*, \hat{t}_j) \qquad \textbf{(EQ 71)}$$

With this definition of $p_i$, we find that agent *i*'s utility is given by:

$$\begin{aligned} u_i(x^*, p_i, t_i) = v_i(x^*, t_i) - p_i &= v_i(x^*, t_i) - \left( \sum_{j \neq i} v_j(x^{-i}, \hat{t}_j) - \sum_{j \neq i} v_j(x^*, \hat{t}_j) \right) \\ &= \left( \sum_{j \neq i} v_j(x^*, \hat{t}_j) + v_i(x^*, t_i) \right) - \sum_{j \neq i} v_j(x^{-i}, \hat{t}_j) \end{aligned} \qquad \textbf{(EQ 72)}$$

The first term (in the parenthesis) can be viewed as the overall utility from social choice $x^*$and the second term as the utility due to the social choice made considering everyone but *i*. Therefore, the VCG mechanism gives agent *i* a utility that corresponds to its own contribution to the overall utility.

**Example 25: (VCG mechanism)**

Consider a company where three departments would like to purchase and share a single enterprise router that costs $3000. The department IT heads get together with the CIO who wants to know whether they really value the router enough to justify having the company spend $3000 on it. If the CIO simply asked the department IT heads (the agents) how much they value the router, they have no incentive to tell the truth, so they would all insist that they needed it. The CIO could, instead, implement a VCG mechanism as follows. Suppose that agent 1 thinks their department's share of the router is worth \$500, agent 2 thinks their department's share is also worth \$500, and agent 3 thinks their department's share of the router is worth \$2500. We represent this as $v_1 = v_2 = 500$; $v_3 = 2500$. Since they sum to more than \$3000, the router should be bought, assuming the agents tell the truth. That is, $x^* =$ 'purchase.'

To ensure truthfulness, the CIO demands payments from each agent (this could be from the departmental IT budget). Assume that the CIO uses the Clarke Pivot payment rule described in Equation 71. We see that $x^{-1} =$ 'purchase', $x^{-2} =$ 'purchase', and $x^{-3} =$ 'do not purchase.' Obviously, $v_i$ is 0 if the decision is 'do not purchase' and the valuation described above if the decision is 'purchase.' This allows us to compute $p_1 = (500 + 2500) - (500 + 2500) = 0$, which is also the same as $p_2$. However, $p_3 = (0) - (500 + 500) = -1000$. In other words, $p_3$ receives a net payment of \$1000 from the CIO! We see that even with the Clarke Pivot value, the VCG mechanism does not achieve budget balance. We do achieve individual rationality: everyone is better off participating in the mechanism than not.

In general, if the non-participation of a single agent can affect the outcome (as it can here), we cannot achieve budget balance with a VCG mechanism. Nevertheless, it is important to note that the VCG mechanism makes truth-telling a dominant strategy, so the CIO can expect that each department head will tell the truth.

[]

Despite its lack of budget balance, the VCG mechanism can be proved to be individually rational, efficient, and strategy proof. Moreover, under some weak conditions (including that no single agent can affect the outcome), the VCG mechanism can also achieve weak budget balance. Therefore, it is the 'gold standard' for dominant-strategy mechanisms and widely used.

### 7.3.8 Problems with VCG mechanisms

The VCG mechanism has two important properties. First, it allows us to design and analyze practical network protocols and algorithms using game theory. It is, therefore, the most engineering-oriented aspect of game theory, thus appealing to computer scientists and engineers. Second, it is remarkable in that it makes agents reveal their true types. This is intimately connected to Byzantine agreement, a classic problem in distributed computer algorithm design. Nevertheless, there are many drawbacks of the VCG approach that we briefly discuss next (for more detail, see [M.H. Rothkopf, "Thirteen reasons the Vickrey-Clarke-Groves process is not practical," Operations Research, 55:2, pp191-197, 2007])

Information requirements: The VCG mechanism assumes that the principal knows the form of the utility function of each agent so that revelation of the agent's type is sufficient to compute its utility. This is a strong assumption. It may not always be possible for principals to know agent utilities, which, after all, reflect their complex inner motivations.

Complexity of valuation function: The computation of the optimal social choice requires the principal to compute the valuation for each agent for each possible alternative. Consider a principal that wants to sell *m* different goods, where players can buy any subset of the goods and value each subset differently. This is called a *combinatorial auction* and may reflect the fact that agents may benefit only from purchasing two or three specific goods, rather than from each good individually. Then, each agent needs to specify up to $2^m$ different values and the principal would need to compute sums over all possible partitions of the *m* goods and their allocation to the agents, an enormous task.

Centralization: The social choice function in a VCG is computed by a single centralized principal who receives inputs from all the agents. Imagine a resource allocation problem with hundreds or thousands of agents: this would require the principal to perform a very large optimization, which is computationally expensive. It would be preferable to have this computation broken up into smaller, distributed computations.

Non-approximability: The VCG mechanism (in dominant strategies, at least) requires that the principal compute the exact optimal value of the sum of the agent valuations. If this is not the optimal value, then agents lose incentive compatibility. However, finding the optimal point of the sum of valuations is complex, and may only be approximable, leaving the mechanism potentially open to manipulation.

Fairness: The VCG scheme assumes that all agents have the same value for money. If this is not true (if, for example, richer agents care less for money than poorer agents), then fairness is not assured.

Budget-balance: We would like any mechanism to be net budget-balanced, so that there are no net payments made to the principal or to the agents. At least, it should not cost the principal money to run the mechanism. However, if a single player can affect the outcome, VCG is not even weakly budget-balanced. It turns out that a different solution concept--Bayes-Nash equilibrium--can guarantee budget balance. The corresponding mechanism, called d'Aspremont-Gerard-Varet (dAGVA) after its inventors, lets each agent compute expected utilities as a function of their prior subjective probabilities on the types of the other agents. However, this budget balance comes at the expense of individual rationality (some agents would be better off not participating in the mechanism).

The solutions to these problems forms the basis of a rapidly growing literature in the field.

## *7.4 Limitations of game theory*

Having studied some aspects of game theory, we now outline some of the limitations of the approach.

- Perhaps the biggest problem with using game theory in real life is ensuring that all players are aware of each others utilities from each outcome. In real life, players often do not know what actions other players are permitted, their payoffs for each outcome, and the utility they gain from these payoffs.
- A second problem has to do with modelling time. A normal form game is played simultaneously by both players and an extensive form game is played sequentially. In neither case, however, do we model the timing of the underlying events. Time and delay are critical factors in most networking problems. For example, in a wireless LAN, a station's transmission is known to others only after a non-trivial delay. Therefore, each player may see a different view of the world at each point in time. This affects the outcome of the game but is not modelled by classical game models.
- Almost all games assume that players are rational. However, there is considerable experimental evidence that people are not rational and sometimes do not even have consistent preferences, undermining utility functions as valid descriptions of user satisfaction.
- Most game models assume that the number of players does not change over time. However, in most typical networks, the number of players (i.e., endpoints sharing a resource) is constantly changing. An endpoint usually does not know who else is sharing a given resource, let alone their utilities and payoffs.
- Any social welfare function that maximizes sums of utilities is implicitly performing inter-personal utility comparisons. This is fundamentally invalid. The standard justification is that all players have the same value for money, but this is certainly invalid.
- As we have seen with mechanism design, games may require massive communication amongst players or between agents and the principal. For instance, to form Bayesian expectations on the other players' types, a player may need to observe their past behaviours. This is both an invasion of privacy and expensive to communicate.

We conclude that it is unlikely that we can use game theory in practice. However, it provides deep insights into modelling the behavior of selfish agents and into the design of communication protocols that can not be manipulated to subvert the designer's intentions. These mitigating factors make it well worth our study.

## *7.5 Exercises*

### 1 Preferences

Suppose that you equally like a banana and a lottery that gives you an apple 30% of the time and a carrot 70% of the time. Also, you equally like a peach and a lottery that gives you an apple 10% of the time and a carrot 90% of the time. (a) What can you say about your relative preferences for bananas and peaches? If you had a lottery whose payoffs were bananas and carrots, what probability of winning a banana or a carrot would be equally preferred to a peach?

### 2 Utility functions

Your cable company gives you 10GB of free data transfer a month, and charges $5/GB thereafter. Suppose that your utility from transferring $x$ GB of data is $100(1-e^{-0.25x})$ and that your disutility from paying $1 is 1. How much data should you transfer in a month to maximize your utility?

### 3 Pure and mixed strategies

Consider the game of tic-tac-toe. What are the possible actions for the first move of the first player (ignore symmetries)? What would constitute a pure strategy? What would constitute a mixed strategy? Would you ever play a mixed strategy for this game? Why or why not?

### 4 Zero-sum game

If the payoffs $(a, -a)$ of every outcome of a zero sum game were changed so that the new payoffs were $(a+5, -5a)$, the game would no longer be zero sum. But, would the structure of the game change?

### 5 Representation

Represent the Pricing game of Example 7 in Normal form.

### 6 Representation

Prove that normal and extensive form are equivalent if information sets are permitted.

### 7 Best response

What is the best response for the customer in the Pricing game (Example 7)?

### 8 Dominant strategy

Suppose that you are not well prepared for a final, and you think you might fail it. If you miss the exam, you will certainly fail it. What is your dominant strategy: attend or miss? Why?

### 9 Bayesian game

Does the Bayesian game in Example 11 have a dominant strategy for the Row player? If so, what is it?

### 10 Repeated game

Suppose that both players in Prisoner's Dilemma (Example 15) play their dominant strategy in an infinitely repeated game with a discount factor of 0.6. What is their payoff for the repeated game?

### 11 Dominant strategy equilibrium

Interpret the meaning of the dominant strategy equilibrium of Example 14. Look up how the 802.11e EDCA protocol solves this problem.

### 12 Iterated deletion

Show an example of a pure strategy that is dominated by a mixture of other pure strategies, although none of the strategies in the mixture dominate the pure strategy.

### 13 Maximin

What are the maximin equilibria in Examples 10 and 14?

### 14 Maximin in a zero-sum game

Show that in Example 18 if Row uses any value of $p$ other than 0.5, then it may get a payoff lower than 2.5 if Column plays either pure or mixed strategies.

### 15 Nash equilibrium

Referring to Example 19, assume that if Column plays a mixed strategy with probability $q$H + (1-$q$)T instead of its Nash equilibrium strategy. What is Row's mixed strategy best response?

### 16 Correlated equilibrium

Does the WiFi game of Example 6 have a correlated equilibrium? If so, describe it.

## 17 Price discrimination

Outline the design of a price-discrimination mechanism with *n* player types (whose valuations are known).

## 18 VCG mechanism

The CIO of a company wants to decide how much capacity to buy from its ISP. The cost of capacity is \$20/Mbps/month. There are three departments in the company, who value capacity as follows: department 1 (D1) values capacity *x*Mbps/month at \$20(*1-$e^{-0.5x}$), D2 values it at \$40(*1-$e^{-0.5x}$), D3 values it at \$80(*1-$e^{-0.5x}$). (a) Assuming the disutility of ISP payment is linear in the amount of payment, what is the overall function that the CIO should maximize? (b) What is type of each department? (c) What is the optimal social choice? (d) What are the Clarke Pivot payments for each department? (e) Is this budget balanced?

# Solutions to Exercises

## A1.1 Probability

### 1 Sample space

The sample space for CW is the discrete set {CWMIN, 2* CWMIN, 4* CWMIN, ...$2^n$*K*CWMIN}, where K is chosen so that $2^n$*K*CWMIN < CWMAX. The sample space for backoff, given CW is a subset of the real line defined by [0, CW].

### 2 Interpretations of probability

An objective interpretation would be that we have a complete weather model that has an intersect source of randomness. Given this model and the current weather conditions, the model predicts that the probability of a snowstorm is 25%.

A frequentist approach would be to look at all prior days where today's weather conditions also held, and look at the number of such days where there was a snowstorm the next morning. We would see that 25% of the time, given the current weather, there was as snowstorm.

A subjective interpretation would be that an expert, who knew all the variables, would take 4:1 odds (or better) on a bet that it would snow tomorrow.

### 3 Conditional probability

(a) We have P(UDP) = 0.2, and P(UDP AND 100) = 0.1. So, P(100 | UDP) = 0.1/0.2 = 0.5.

(b) Here, P(UDP) = 0.5 and P(100|UDP) = 0.5. So, P(100 AND UDP) = 0.5*0.5 = 0.25.

### 4 Conditional probability again

Before you know the protocol type of a packet, the sample space is all possible packet lengths of all possible protocol types. After you know the protocol type, the sample space only include packet lengths for that protocol.

## 5 Bayes’ rule

P(UDP|100) = (P(100|UDP)P(UDP))/P(100). We need P(100) = $x$. Then, P(UDP|100) = 0.5*0.2/$x$ = 0.1/$x$.

## 6 Cumulative distribution function

(a) $F_D(i) = \sum_{j=1}^{i} \frac{1}{2^j} = 1 - 2^{-i}$.

(b) $f_{C(x)} = \frac{1}{x_2 - x_1}$, so $F_C(x) = \int_{x_1}^{x} \frac{1}{x_2 - x_1} dx = \frac{x - x_1}{x_2 - x_1}$.

## 7 Expectations

(a) $E[D] = \sum_{j=1}^{i} \frac{i}{2^j}$.

(b) By geometry, E[C] = $(x_2+x_1)/2$ (you can also derive this analytically).

## 8 Variance

$V[aX] = E[a^2X^2] - (E[aX])^2 = a^2(E[X^2] - (E[X])^2) = a^2V[X]$.

## 9 Bernoulli distribution

Consider the event E defined as ‘Room X is making an outgoing call during the busy hour.’ Clearly, P(E) =$p$ = 1/6. The probability of 5 simultaneous calls is $\binom{20}{5}\left(\frac{1}{6}^5\right)\left(\frac{5}{6}\right)^{15} = 0.129$ and of 15 simultaneous calls is $\binom{20}{15}\left(\frac{1}{6}^{15}\right)\left(\frac{5}{6}\right)^{5} = 1.33*10^{-8}$ .

## 10 Geometric distribution

Packet and ack transmissions are geometrically distributed with parameter $p$=0.9. So the expected number of packet transmissions is 1/$p$ = 1.11 and the expected number of ack transmissions is also 1.11. These are independent events, so the expected number of data transmissions for successful packet+ack transfer = 1.11+1.11 = 2.22.

## 11 Poisson distribution

(a) Using the binomial distribution, the value is $\binom{10}{8}(0.1^8)(0.9^2)$ = .36*10$^{-6}$. For the Poisson approximation, λ= 1, so the value is $P(X = 8) = e^{-1}\left(\frac{1^8}{8!}\right)$ = 9.12*10$^{-6}$. (b) Using the binomial distribution, the value is $\binom{100}{8}(0.1^8)(0.9^{92})$ = .114. For the Poisson approximation, λ= 10, so the value is $P(X = 8) = e^{-10}\left(\frac{10^8}{8!}\right)$ = .112. It is clear that as $n$ increases, the approximation greatly improves.

## 12 Gaussian distribution

Consider the cumulative distribution of $Y = F_Y(y) =$

$P(Y \le y) = P(aX + b \le y) = P\left(X \le \frac{(y-b)}{a}\right) = F_X\left(\frac{(y-b)}{a}\right)$ if a > 0. Then, $f_Y(y) = F'_Y(y) =$

$$F_X'\left(\frac{(y-b)}{a}\right) = \frac{1}{a}f_x\left(\frac{(y-b)}{a}\right) = \frac{1}{a\sigma\sqrt{2\pi}}e^{-\frac{\left(\left(\frac{y-b}{a}\right)-\mu\right)^2}{2\sigma^2}} = \frac{1}{a\sigma\sqrt{2\pi}}e^{-\frac{(y-b-a\mu)^2}{2a^2\sigma^2}} = \frac{1}{a\sigma\sqrt{2\pi}}e^{-\frac{(y-(b+a\mu))^2}{2a^2\sigma^2}} .$$

Comparing with the standard definition of a Gaussian, we see that the parameters of $Y$ are ($a\mu + b$ , $(a\sigma)^2$ ). A similar calculation holds if $a < 0$.

### 13 Exponential distribution

We have $1/\lambda$ =5. We need to compute 1-$F$(15) = $1-(1-e^{-\lambda x}) = e^{\frac{-15}{5}} = e^{-3} = 4.98$ %.

### 14 Exponential distribution

Because the exponential distribution is memoryless, the expected waiting time is the same, i.e. 200 seconds, no matter how long your break for icecream. Isn't that nice?

### 15 Power law

| $x$ | $f_{power\_law}(x)$ | $f_{exponential}(x)$ |
|---|---|---|
| 1 | 1 | 0.27 |
| 5 | 0.04 | $9.07*10^{-5}$ |
| 10 | 0.01 | $4.1*10^{-9}$ |
| 50 | $4*10^{-4}$ | $7.44*10^{-44}$ |
| 100 | $1*10^{-4}$ | $2.76*10^{-87}$ |

It should now be obvious why a power-law distribution is called 'heavy-tailed'!

### 16 Markov's inequality

(a) We need 1-$F(10) = e^{-20}$= $2.06*10^{-9}$. (b) The mean of this distribution is 1/2. So, $P(X \geq 10) \leq \frac{0.5}{10} = 0.05$ . It is clear that the bound is very loose.

### 17 Joint probability distribution

(a) $p_X$= {0.5, 0.5}; $p_Y$= {0.2, 0.8}; $p_Z$= {0.3, 0.7}; $p_{XY}$= {0.1, 0.4, 0.1, 0.4}; $p_{XZ}$= {0.15, 0.35, 0.15, 0.35}; $p_{YZ}$= {0.1, 0.1, 0.2, 0.6}

(b) $X$ and $Y$ are independent because $p_{XY} = p_X p_Y$. $X$ and $Z$ are independent because $p_{XZ} = p_X p_Z$.

(c) P($X$=0|$Z$=1) = P($X$=0 AND $Z$=1)/P($Z$=1) = 0.35/0.7 = 0.5.

## A1.2 Statistics

### 1 Moments

$$\mu_3 = E((X-\mu)^3) = E(X^3 - 3X^2\mu + 3X\mu^2 - \mu^3) = \mu'_3 - 3\mu E(X^2) + 3\mu^2 E(X) - \mu^3 = \mu'_3 - 3\mu\mu_2' + 3\mu^3 - \mu^3$$

### 2 MGFs

$$\int_0^1 \left(1 + tx + \frac{(tx)^2}{2!} + \frac{(tx)^3}{3!} + \ldots\right) dx = x\Big|_0^1 + \frac{tx^2}{2!}\Bigg|_0^1 + \frac{t^2x^3}{3!}\Bigg|_0^1 + \frac{t^3x^4}{4!}\Bigg|_0^1 + \ldots$$

$$= 1 + \frac{t}{2!} + \frac{t^2}{3!} + \frac{t^3}{4!} + \ldots = \frac{1}{t}\left(t + \frac{t^2}{2!} + \frac{t^3}{3!} + \ldots\right) = \frac{1}{t}\left(1 + t + \frac{t^2}{2!} + \frac{t^3}{3!} + \ldots - 1\right) = \frac{1}{t}(e^t - 1)$$

## 3 MGFs

To find the *r*th moment, we differentiate the MGF for the uniform distribution, i.e. $\frac{1}{t}(e^t - 1)$ *r* times and then set *t* to zero. Working directly from the series, we need to differentiate the expression $1 + \frac{t}{2!} + \frac{t^2}{3!} + \frac{t^3}{4!} + \ldots$ *r* times and set *t* to 0. Note that all terms with powers of *t* smaller than *r* disappear when we differentiate this series *r* times. Moreover, all terms with powers of *t* greater than *r* disappear when we set *t* to zero after differentiation (why?). Therefore, the only term we need to consider the $t^r/(r+1)!$. It is clear the when we differentiate this *r* times, we get the term $r!/(r+1)!$, which reduces to $1/1+r$ as stated.

## 4 MGF of a sum of two variables

The MGF of the sum of two independent uniform random variables $X_1$ and $X_2$ is $\frac{1}{t^2}[e^t - 1]^2$ (from Example 3), so, the MGF of $(X\text{-}\mu)$ is given by $\frac{e^{-\mu t}}{t^2}[e^t - 1]^2$. To find the variances need to differentiate this expression twice with respect to *t* and then set *t* to 0. Given the *t* in the denominator, it is convenient to rewrite the expression as $\left(1 - \mu t + \frac{\mu^2 t^2}{2!} + \ldots\right)\left(1 + \frac{t}{2!} + \frac{t^2}{3!} + \ldots\right)\left(1 + \frac{t}{2!} + \frac{t^2}{3!} + \ldots\right)$ (we have divided $e^t$-1 by *t* in each of the second and third terms), where the ellipses refer to terms with third and higher powers of *t*, which will reduce to 0 when *t* is set to 0. In this product, we need only consider the coefficient of $t^2$, which is $\frac{\mu^2}{2!} + \frac{1}{3!} + \frac{1}{3!} - \frac{\mu}{2!} - \frac{\mu}{2!} + \frac{1}{2!2!}$. Differentiating the expression twice results in multiplying the coefficient by 2. Note that for the sum of two uniform standard random variables, $\mu = 1$, so that when we set *t* to zero, we obtain $E((X\text{-}\mu)^2) = V(X) = 2\left(\frac{1}{2} + \frac{1}{6} + \frac{1}{6} - \frac{1}{2} - \frac{1}{2} + \frac{1}{4}\right) = \frac{1}{6}$. As a check, note that the variance of each variable is 1/12, so that the variance of the sum is the sum of the variances, as we found.

## 5 MGF of a normal distribution

The MGF of $a+bX$ is $e^{at}M(bt) = e^{at}e^{\mu bt + \frac{1}{2}\sigma^2(bt)^2} = e^{(a+\mu b)t + \frac{1}{2}(\sigma^2 b^2)t^2}$.Set $a = \frac{-\mu}{\sigma}$ and $b = \frac{1}{\sigma}$. Then, $e^{(a+\mu b)t + \frac{1}{2}(\sigma^2 b^2)t^2} = e^{\left(\frac{-\mu}{\sigma} + \frac{\mu}{\sigma}\right) + \frac{1}{2}\left(\frac{\sigma^2}{\sigma^2}\right)t^2} = e^{\frac{t^2}{2}}$, which is the MGF of a *N(0,1)* variable.

## 6 Means

To minimize $\sum_{i=1}^{n}(x_i - x^*)^2$, we differentiate the expression with respect to $x^*$ and set this value to 0. We find that $\frac{d}{dx^*}\sum_{i=1}^{n}(x_i - x^*)^2 = \sum_{i=1}^{n} -2(x_i - x^*) = 0$, so that $\sum x_i - \sum x^* = 0$. Rewriting $\sum x^*$ as $nx^*$, we get the desired result.

## 7 Means

$$\frac{1}{n}\left(\sum_{i=1}^{n}(x_i-\mu)^2 - n(\bar{x}-\mu)^2\right) = \frac{1}{n}\left(\sum_{i=1}^{n}(x_i^2+\mu^2-2x_i\mu) - n(\bar{x}^2+\mu^2-2\bar{x}\mu)\right)$$

$$= \frac{1}{n}\left(\sum_{i=1}^{n}x_i^2+\mu^2-2x_i\mu-\bar{x}^2-\mu^2+2\bar{x}\mu\right) = \frac{1}{n}\left(\sum_{i=1}^{n}x_i^2-2x_i\mu-\bar{x}^2+2\bar{x}\mu\right) = \frac{1}{n}\left(\sum_{i=1}^{n}x_i^2-\bar{x}^2+\sum_{i}2\bar{x}\mu-2x_i\mu\right)$$ . But

$\sum_{i}(2\bar{x}\mu-2x_i\mu) = 2n\bar{x}\mu-2n\bar{x}\mu = 0$, hence the desired result.

## 8 Confidence intervals (normal distribution)

The sample mean is 61.11. We compute $\sum_{i=1}^{n}(x_i-\bar{x})^2$ as 936647.76. Therefore, the variance of the sampling distribution of the mean is estimated as 936647.76/(17*16) = 3443.55 and the standard deviation of this distribution is estimated as its square root, i.e., 58.68. Using the value of ±1.96σ for the 95% confidence interval, the 95% confidence interval is 61.11±115.02. The very large interval is due to the outlier value.

## 9 Confidence intervals (t distribution)

We simply substitute the value of ±2.12σ to obtain the interval as 61.11±124.40.

## 10 Hypothesis testing: comparing the mean to a constant

The mean is 2.46%. We compute the variance as 0.0076% and the standard deviation as 0.87%. We could use the t distribution to test the hypothesis, but it is clear by inspection that 2% lies within 1 standard deviation of the mean, so we cannot reject the null hypothesis. For completeness' sake, the confidence interval for the t distribution with 9 degrees of freedom (at the 95% level) is 2%±2.262*0.87%.

## 11 Chi-squared test

The critical value of $n_1$ is when the chi-squared value is $X = (n_1\text{-}42)^2/42 + (100\text{-}n_1\text{-}58)^2/58 = 3.84$. Solving, we get $n_1 > 51.67$. So, an value greater than or equal to 52 will result in the hypothesis being rejected.

## 12 Fitting a distribution and chi-squared test

The total number of time periods is 28+56+...+5 = 1193. The total number of arrivals is (28*2)+(56*3)+...+(5*16) = 8917. Therefore, the mean number of packets arriving in 1ms is 8917/1203 = 7.47. This is the best estimate of the mean of a Poisson distribution. We use this to generate the probability of a certain number of arrivals in each 1ms time period using the Poisson distribution. This probability multiplied by the total number of time periods is the expected count for that number of arrivals, and this is shown below.

| **Number of packet arrivals** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Count | 28 | 56 | 105 | 126 | 146 | 164 | 165 | 120 | 103 | 73 | 54 | 23 | 16 | 9 | 5 |
| Expected count | 19 | 47 | 88 | 132 | 164 | 175 | 164 | 136 | 102 | 69 | 43 | 25 | 13 | 7 | 3 |

The chi-squared value is computed as $(28\text{-}19)^2/21 + (56\text{-}47)^2/47 .... + (5\text{-}3)^2/3 = 19.98$. Since we estimated one parameter from the sample, the degrees of freedom = 15-1-1 = 13. From the chi-squared table, with 13 degrees of freedom, at the 95% confidence level, the critical value is 22.36. Therefore, we cannot reject the hypothesis that the sample is well-described by a Poisson distribution at this confidence level.

## 13 Independence, Regression, and Correlation

(a) If number of peers were independent of the number of peers, then, as the uplink capacity changed, the number of peers should remain roughly constant and equal to the population mean, whose best estimate is the sample mean. Therefore, the expected value of the number of peers is 50+31+...+49/10 = 40.4.

(b) The chi-squared variate is $(50\text{-}40.4)^2/40.4 + (31\text{-}40.4)^2/40.4 +... + (49\text{-}40.4)^2/40.4 = 27.93$. Because we estimated one parameter from the data set (i.e., the mean), we have 10-1-1 = 8 degrees of freedom. We find that at the 95% confidence level, the critical value of the chi-squared distribution with 8 degrees of freedom is 15.51. Therefore, we can reject the hypothesis that the number of peers is independent of the uplink capacity with 95% confidence. The critical value at the 99.9% level is 25.125, so we can reject the hypothesis even at the 99.9% level.

(c) We use the equation $b = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})^2}$ to find $b = 0.21$.

(d) Using $r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\left(\sum (x_i - \bar{x})^2\right)\left(\sum (y_i - \bar{y})^2\right)}}$ , we find $r = 0.952$, which is close to 1. Therefore, we can state that the two variables are well-represented by a linear relationship, which indicates dependence, rather than independence.

(e) The portion of variability in the number of peers by the uplink capacity is $r^2 = 90.1\%$.

### 14 Correlation coefficient

For convenience, we use the following notation:

$$X^2 = \sum (x_i - \bar{x})^2$$

$$Y^2 = \sum (y_i - \bar{y})^2$$

$$XY = \sum (x_i - \bar{x})(y_i - \bar{y})$$

$S^2 = \Sigma(y_i - a - bx_i)^2$. Ignoring the summation symbol, we can rewrite the summand as

$$\left(y_i - \left(\bar{y} - \bar{x}\frac{XY}{X^2}\right) - x_i\frac{XY}{X^2}\right)^2 = \left((y_i - \bar{y}) - \frac{(x_i - \bar{x})XY}{X^2}\right)^2 = Y^2 + \frac{(XY)^2X^2}{X^2X^2} - 2\frac{(XY)^2}{X^2} = Y^2 - \frac{(XY)^2}{X^2} = Y^2 - Y^2\left(\frac{(XY)^2}{Y^2}\right)$$

$= Y^2(1 - r^2)$ , as desired. To understand the third step, recall the presence of the summation symbol.

### 15 Single Factor ANOVA

Here $I = 3$ and $J = 10$. We compute $\overline{Y_{1.}} = 55.53$, $\overline{Y_{2.}} = 55.94$, $\overline{Y_{3.}} = 55.95$. This allows us to compute $SSW = 3102.29$ and $SSB = 1.15$. The $F$ statistic is therefore (1.15/ 2)/(3102.29/27) = 0.0050. Looking up the $F$ table we find that with (3, 27) degrees of freedom, the critical $F$ value even at the 5% confidence level is 2.96. The computed statistic is far below this value. Therefore, the null hypothesis cannot be rejected.

## A1.3 Linear Algebra

### 1 Transpose

$$\begin{bmatrix} 4 & 7 & 3 \\ 0 & 82 & -2 \\ -3 & 12 & 2 \end{bmatrix}$$

## 2 Matrix multiplications

$$\begin{bmatrix} -44 & 118 & -54 \\ 59 & 14 & -24 \\ -40 & 20 & 40 \end{bmatrix}$$

## 3 Exponentiation

The proof is by induction. The base case is for *k=2*, where by direct computation, we find that

$\begin{bmatrix} a_{11} & \cdots & 0 \\ \cdots & a_{ii} & \cdots \\ 0 & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} a_{11} & \cdots & 0 \\ \cdots & a_{ii} & \cdots \\ 0 & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} a_{11}^2 & \cdots & 0 \\ \cdots & a_{ii}^2 & \cdots \\ 0 & \cdots & a_{nn}^2 \end{bmatrix}$. The inductive assumption is that

$A^k = \begin{bmatrix} a_{11} & \cdots & 0 \\ \cdots & a_{ii} & \cdots \\ 0 & \cdots & a_{nn} \end{bmatrix}^k = \begin{bmatrix} a_{11}^k & \cdots & 0 \\ \cdots & a_{ii}^k & \cdots \\ 0 & \cdots & a_{nn}^k \end{bmatrix}$. Then, we compute the $k+1^{\text{th}}$ power of $A$ as

$A^k A = \begin{bmatrix} a_{11}^k & \cdots & 0 \\ \cdots & a_{ii}^k & \cdots \\ 0 & \cdots & a_{nn}^k \end{bmatrix} \begin{bmatrix} a_{11} & \cdots & 0 \\ \cdots & a_{ii} & \cdots \\ 0 & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} a_{11}^{k+1} & \cdots & 0 \\ \cdots & a_{ii}^{k+1} & \cdots \\ 0 & \cdots & a_{nn}^{k+1} \end{bmatrix}$. QED.

## 4 Linear combination of scalars

The linear combination is 10*0.5 + 5*0.4 + 2*0.25 + -4*0.25 = 5 + 2 + .5 - 1 = 6.5

## 5 Linear combination of vectors

The first element of the linear combination is given by 1*0.5 + 3*0.4 + 7*0.25 + 2*0.25 = 0.5 + 1.2 + 1.75 + 0.5 = 3.95. Computing the other elements similarly, we obtain the solution [3.95 5.8 6.2 6.15].

## 6 Linear independence and rank.

The implicitly defined coefficient matrix is given by

$\begin{bmatrix} 12 & 2 & -4 \\ 2 & 2 & -24 \\ 2.5 & 0 & 5 \end{bmatrix}$. If the vectors are independent, then the rank of this matrix will be 3, so that Gaussian elimination would result in no equations being reduced to the trivial form 0=0. We proceed with Gaussian elimination as follows: Equation 3 does not contain the second variable, so we remove the second variable from the second equation by subtracting the first row from the second row, to get $\begin{bmatrix} 12 & 2 & -4 \\ -10 & 0 & -20 \\ 2.5 & 0 & 5 \end{bmatrix}$. It is clear that the second row is the third row multiplied by -4, so that if we add 4 times the third row to the second row, we get $\begin{bmatrix} 12 & 2 & -4 \\ 0 & 0 & 0 \\ 2.5 & 0 & 5 \end{bmatrix}$. Can we reduce any of the remaining equations to the form 0=0? It is clear that the first row is not a multiple of the third row, because the second element of the third row is 0, and the second element of the first row is not. Hence, the rank of the co-efficient matrix is 2, which is smaller than 3, so that the three vectors are *not* independent.

## 7 Basis and dimension

Two of the three vectors are linearly independent, so we have two vectors in the basis and a generated vector space of dimension 2. One possible basis is simply the two vectors themselves, that is, {[12 2 -4], [2.5 0 5]}. We can get another basis by multiplying either vector by any scalar. For example, we can multiply the first vector by 0.5 to get another basis as {[6 1 -2], [2.5 0 5]}.

## 8 Gaussian elimination

Noticing that the third equation has a zero in the second column, we will eliminate the second variable in the firs row as well, by subtracting twice the second row from the first row, to obtain $\begin{bmatrix} 22 & 0 & -16 & 9 \\ -8 & 2 & 4 & -2 \\ 10 & 0 & 4 & 1 \end{bmatrix}$. We can eliminate the third variable from the first row by multiplying the third row by 4 and adding it to the first row, to get $\begin{bmatrix} 62 & 0 & 0 & 13 \\ -8 & 2 & 4 & -2 \\ 10 & 0 & 4 & 1 \end{bmatrix}$. We can read off $x_1$ = 13/62 = 0.2096. Substituting in the third row, we find 2.096 + 4$x_3$ = 1, so that $x_3$ = (1-2.096)/4 = -0.274. Substituting these in the first row of the original equation, we find 6*0.2096 + 4*$x_2$ -8*-0.274 = 5, so that $x_2$ = 0.3876.

## 9 Rank

Consider the *i*th row of a non-zero diagonal matrix. Its diagonal element is $a_{ii}$ which is not 0, but all other elements in the *i*th column are 0. Therefore, there is no way to obtain the *i*th row as a linear combination of the other rows. Since this is true for all *i*, the rows are all linearly independent, and the rank of the matrix is *n*. Note that the rows are therefore a basis of the corresponding vector space.

## 10 Determinant

Expanding by the second column, we find the determinant to be 8*(4*2 - 3*(-3)) - (-2)*(4*12-7*(-3)) = 8*(8+9) +2*(48+21) = 8*17+2*69 = 274.

## 11 Inverse

We already know the determinant of the matrix is 274 (see Exercise 10). The co-factor $C_{11}$ is given by (-1)$^{1+1}$(8*2-(-2)*(12)) = 16+24 = 40. $C_{21}$= (-1)$^{1+2}$(0*2 -(-2)*(-3)) = -(-6)= 6. Computing the other co-factors similarly, we obtain the inverse as $\frac{1}{274}\begin{bmatrix} 40 & 6 & 24 \\ 22 & 17 & -69 \\ -38 & 8 & 32 \end{bmatrix}$.

## 12 Matrix as a transformation

Let the angle made by the vector from (0,0) to (*x, y*) be *t* and let its length be *r*. Then, we can write *x* and *y* as

$$x = r\cos(t)$$

$$y = r\sin(t)$$

Let the rotated vector join the origin to the point (*X,Y*). We expand:

$$X = r\cos(t+p) = r(\cos(t)\cos(p) - \sin(t)\sin(p)) = x*\cos(p) - y*\sin(p)$$

$$Y = r\sin(t+p) = r(\sin(t)\cos(p) + \cos(t)\sin(p)) = y*\cos(p) + x*\sin(p)$$

We can write this as

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} \cos(p) & -\sin(p) \\ \sin(p) & \cos(p) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

so that the rotation matrix is $\begin{bmatrix} \cos(p) & -\sin(p) \\ \sin(p) & \cos(p) \end{bmatrix}$.

## 13 Composing transformations

We compute the composition as

$$\begin{bmatrix}\cos(p) & -\sin(p)\\ \sin(p) & \cos(p)\end{bmatrix}\begin{bmatrix}\cos(t) & -\sin(t)\\ \sin(t) & \cos(t)\end{bmatrix} = \begin{bmatrix}\cos(p)\cos(t)-\sin(p)\sin(t) & -\cos(p)\sin(t)-\sin(p)\cos(t)\\ \sin(p)\cos(t)+\cos(p)\sin(t) & -\sin(p)\sin(t)+\cos(p)\cos(t)\end{bmatrix} =$$

$\begin{bmatrix}\cos(p+t) & -\sin(p+t)\\ \sin(p+t) & \cos(p+t)\end{bmatrix}$, which we recognize as a rotation by a total of *t*+*p* degrees, as expected.

## 14 Eigenvalues and eigenvectors

The characteristic equation is $\begin{vmatrix}1-\lambda & 9\\ 4 & 1-\lambda\end{vmatrix} = 0$, so that $(1-\lambda)^2 - 36 = 0$, and we get $\lambda = -5, 7$ as the eigenvalues.

We compute the eigenvector corresponding to the value -5 by solving the equation

$\begin{bmatrix}1 & 9\\ 4 & 1\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} = (-5)\begin{bmatrix}x_1\\ x_2\end{bmatrix}$. This gives us the equations $x_1 + 9x_2 = -5x_1$; $4x_1 + x_2 = -5x_2$. Either one can be solved to get $x_2 = -(2x_1)/3$, corresponding to an eigenvector family of scalar multiples of $[\ 1\ -2/3]^T$.

We compute the eigenvector corresponding to the value 7 by solving the equation

$\begin{bmatrix}1 & 9\\ 4 & 1\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} = 7\begin{bmatrix}x_1\\ x_2\end{bmatrix}$. This gives us the equations $x_1 + 9x_2 = 7x_1$; $4x_1 + x_2 = 7x_2$. Either one can be solved to get $x_2 = 2x_1/3$, corresponding to an eigenvector family of scalar multiples of $[\ 1\ 2/3]^T$.

## 15 Computing *A^n^x*

From Exercise 14, we know that the eigenvectors of $\begin{bmatrix}1 & 9\\ 4 & 1\end{bmatrix}$ are $\begin{bmatrix}1\\ \frac{2}{3}\end{bmatrix}$ and $\begin{bmatrix}1\\ \frac{-2}{3}\end{bmatrix}$. We recognize the vector $\begin{bmatrix}8\\ 0\end{bmatrix}$ can be written as $4\begin{bmatrix}1\\ \frac{2}{3}\end{bmatrix} + 4\begin{bmatrix}1\\ \frac{-2}{3}\end{bmatrix}$. Hence,

$$\begin{bmatrix}1 & 9\\ 4 & 1\end{bmatrix}^5\begin{bmatrix}8\\ 0\end{bmatrix} = \begin{bmatrix}1 & 9\\ 4 & 1\end{bmatrix}^5\left(4\begin{bmatrix}1\\ \frac{2}{3}\end{bmatrix} + 4\begin{bmatrix}1\\ \frac{-2}{3}\end{bmatrix}\right) = 4\begin{bmatrix}1 & 9\\ 4 & 1\end{bmatrix}^5\begin{bmatrix}1\\ \frac{2}{3}\end{bmatrix} + 4\begin{bmatrix}1 & 9\\ 4 & 1\end{bmatrix}^5\begin{bmatrix}1\\ \frac{-2}{3}\end{bmatrix} = 4\left(-5^5\begin{bmatrix}1\\ \frac{-2}{3}\end{bmatrix} + 7^5\begin{bmatrix}1\\ \frac{2}{3}\end{bmatrix}\right) = \begin{bmatrix}54728\\ 53152\end{bmatrix}$$

## 16 Finding eigenvalues

The matrix is symmetric, so its eigenvalues are real. From the Gerschgorin circle theorem, the eigenvalues lie in the intersection of the real intervals [4-1.5  4+1.5], [6-1.3 6+1.3], [5-0.8  5+0.8] = {[2.5 5.5], [4.7 7.3], [4.2 5.8]} = [2.5 7.3].

## 17 Power method

We start with the initial vector $x_0 = \begin{bmatrix}1\\ 1\end{bmatrix}$. Applying the matrix once, we get $x_1 = \begin{bmatrix}10\\ 5\end{bmatrix}$. The Rayleigh ratio evaluates to( [1 1] * $\begin{bmatrix}10\\ 5\end{bmatrix}$ )/([ 1 1] * $\begin{bmatrix}1\\ 1\end{bmatrix}$ ) = 15/2 = 7.5. Repeating, we get $x_2 = \begin{bmatrix}55\\ 45\end{bmatrix}$ and the ratio evaluates to 775/125 = 6.2. After one more iteration, we get $x_3 = \begin{bmatrix}460\\ 265\end{bmatrix}$, and the ratio evaluates to 37225/5050 = 7.37. For the fourth iteration, we get $x_4 = \begin{bmatrix}2845\\ 2105\end{bmatrix}$, and the ratio evaluates to 1866525/281825 = 6.622. We see that the series slowly converges to the dominant eigenvalue of 7.

To compute the dominant eigenvalue, we start with $x_1 = \begin{bmatrix} 10 \\ 5 \end{bmatrix}$, which we rescale to $x_1 = \begin{bmatrix} 1 \\ 0.5 \end{bmatrix}$. Then, $x_2 = \begin{bmatrix} 1 & 9 \\ 4 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0.5 \end{bmatrix} = \begin{bmatrix} 5.5 \\ 4.5 \end{bmatrix}$, which we rescale to $\begin{bmatrix} 1 \\ 0.818 \end{bmatrix}$. Thus, $x_3 = \begin{bmatrix} 1 & 9 \\ 4 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0.818 \end{bmatrix} = \begin{bmatrix} 8.362 \\ 4.818 \end{bmatrix}$, which we rescale to $\begin{bmatrix} 1 \\ 0.576 \end{bmatrix}$.Finally, $x_4 = \begin{bmatrix} 1 & 9 \\ 4 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0.576 \end{bmatrix} = \begin{bmatrix} 6.184 \\ 4.576 \end{bmatrix}$, which we rescale to $\begin{bmatrix} 1 \\ 0.734 \end{bmatrix}$ and is the estimate of the dominant eigenvector. Compare this to the true value of $x = \begin{bmatrix} 1 \\ 0.66 \end{bmatrix}$.

### 18 Diagonalization

This is the matrix with the same eigenvalues as the given matrix, i.e., $\begin{bmatrix} -5 & 0 \\ 0 & 7 \end{bmatrix}$.

### 19 Stochastic matrix

The matrix is left- (or column-) stochastic but not right- (or row-) stochastic because its columns add to 1.0, but its rows do not.

### 20 State transitions

The initial state vector is [0.5 0.5 0] $^T$. After one time step, the state vector is

$\begin{bmatrix} 0.25 & 0.1 & 0 \\ 0.5 & 0.9 & 0 \\ 0.25 & 0 & 1.0 \end{bmatrix} \begin{bmatrix} 0.5 \\ 0.5 \\ 0 \end{bmatrix} = \begin{bmatrix} 0.175 \\ 0.7 \\ 0.125 \end{bmatrix}$. After another time step, the state vector is $\begin{bmatrix} 0.25 & 0.1 & 0 \\ 0.5 & 0.9 & 0 \\ 0.25 & 0 & 1.0 \end{bmatrix} \begin{bmatrix} 0.175 \\ 0.7 \\ 0.125 \end{bmatrix} = \begin{bmatrix} 0.11375 \\ 0.7175 \\ 0.16875 \end{bmatrix}$. Therefore, the probability of being in state 1 after two time steps is 0.11375, and of being in state 2 after two time steps is 0.7175.

## *A1.4 Optimization*

### 1 Modelling

This problem has many solutions. Here is one possible

Control variables (these are from the statement of the problem):

- *x*i : starting point of flight i
- *di* : duration of flight *i*, $d_i \geq 15$
- The cost of ticket for the *i*th flight, *t*i.

Note that the number of passengers in a flight is not a control parameter, because it is related to the cost of a ticket - once the cost of the ticket is determined, the number of passengers cannot be independently controlled.

Fixed parameters:

- The possible take-off locations, *V* .
- The cost of chase vehicle (gas needed, maintenance, etc.) per kilometer of travel.
- Location of the roads, *R*.
- The cost of the natural gas, *g*, per minute of flight.
- Pilot's wages, *w*, per minute of flight.

Input parameters:

- The wind speed and direction for flight *i*

Transfer functions:

- Where a balloon lands, as a function of starting point, wind speed and direction, and flight duration.
- The number of passengers p as a function of cost of a ticket.
- A function that, given the cost of every process in the business, computes the cost of flight i.

Output parameters:

- For flight *i*, the distance of the balloon landing spot from a road
- Number of passengers for flight *i*, $p_i$
- The cost of flight i, denoted $f_i$

Objective function:

- Maximize $p_i t_i - f_i$

Empirically estimating the transfer functions:

- For each starting point, for each wind speed and direction, empirically determine the landing spot.
- The number of passengers for a cost can be determined by trial and error or by doing a market study.
- The cost of flight can be determined empirically as a linear combination of the xed parameters.

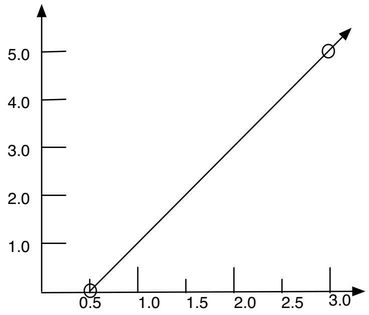
## 2 Optimizing a function of two variables

The figure below shows the plot for the curve $2x_1 - x_2 = 1$ subject to $x_1 \geq 0$ and $x_2 \geq 0$. The system does not impose any constraint on how much $x_1$ and $x_2$ can grow, so the maximum value is unbounded. We know that the minimal value is at a vertex, in this case we only have one (0:5; 0). If we evaluate the function in (0:5; 0) and at a randomly chosen point, say (3; 5) we get:

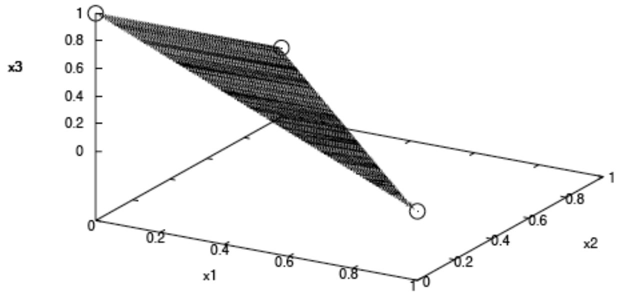
$O(0{:}5; 0) = 5$

$O(3; 5) = 15$

Using this information, we know that the minimum value of O, given the constraints, is 5 at (0:5; 0). There is no maximum value, since O is unbounded in the space dened by the constraints.



## 3 Optimizing a function of three variables



The figure above shows the plot for the plane $x_1 + x_2 + x_3 = 1$ for $x_1, x_2, x_3$ non-negative. The resulting polyhedron serves to find the optimal values of O. The optimal value has to be at a vertex, so we evaluate the value of the objective function at the three vertices.

$O(1; 0; 0) = 5$

$O(0; 1; 0) = 2$

$O(0; 0; 1) = -1$

Clearly, the maximum value is reached at point (1; 0; 0) and the minimum value is reached at point (0; 0; 1).

## 4 Network flow

We will consider the problem when there is only one source node *s*. Otherwise, if we have many sources, we can always create a new source with unbounded capacity to transfer to all the sources, which would each have limited capacity. Similarly, we can unify all the sinks to form a single sink *t*.

Let $G = (V, E)$ be the graph, and let $f_{ij}$ be the flow between nodes $v_i$ and $v_j$. Let $c_{ij}$be the capacity of link from $v_i$ to $v_j$. The classical problem is stated as:

$$O = \sum_i f_{si} \quad \text{subject to}$$

$$\sum_i f_{ij} = \sum_k f_{jk} \quad \forall j \notin \{s, t\}$$

$$f_{ij} \le c_{ij} \quad \forall i, j$$

We can interpret the ‘capacity of a warehouse’ in two ways. One way to interpret it is that no more than $cap_j$ flow can go through warehouse *j*. To model this, we add the following constraint:

$$\sum_i f_{ij} \le cap_j \quad \forall j$$

A more complex interpretation of the constraint is that each warehouse has a limited storage capacity. This would allow the ingress flow to exceed the egress flow for a limited duration of time. Specifically, if the storage capacity of warehouse *j* is $B_j$, then, denoting the flow on link $v_i$-$v_j$ by $f_{ij}(t)$,

$$\sum_i \int_{t_i}^{t_2} f_{ij}(t)dt \le \sum_k \int_{t_i}^{t_2} f_{jk}(t)dt + B_j \quad \forall j \notin \{s, t\}, \forall t_i, \forall t_2$$

so that the ingress flows to any node *j*, integrated over all possible time periods, never exceeds the egress flows, integrated over the same time period, taking into account the possibility of storing $B_j$ units in the warehouse.

### 5 Integer linear programming

Let the variable $x_{ijh}$ indicate whether or not user *i* can schedule a job on machine *h* at time period *j*. Let the cost and benefit of assigning machine *h* to user *i* at time period *j* be $c_{ijh}$and $g_{ijh}$respectively. Then, the function to optimize is

$$O = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{h=1}^{k}(g_{ijh}-c_{ijh})x_{ijh}$$

The obvious constraints are that $x_{ijh} \in \{0, 1\}$, which makes the problem an ILP. In addition, we express the constraint that at each time slot a machine can be assigned to at most one user using the constraint:

$$\sum_{i=1}^{n} x_{ijh} \le 1 \quad \forall j, h$$

### 6 Weighted bipartite matching

The standard bipartite matching allows us to place only one ball in one urn, so we modify the elements of the bipartite graph as follows: given *m* urns in *M* indexed by *i*, we create a new set *M'* that contains 2*m* elements, labelled $m'_{i1}$ and $m^{'}_{i2}$. Now, create links from *K* to *M'* where the payoff on the link from ball $k_j$ to urn $m^{'}_{i1}$ and $m^{'}_{i2}$ is the same as the payoff from ball $k_j$ to urn $m_i$ in the original problem, i.e., $p_{ji}$. The solution to the weighted bipartite matching problem in $M^{'}$ trivially gives us the solution to the problem in *M*.

### 7 Dynamic programming

Let *D*(*i,j*) denote the numbers of errors in a match that between the first *i* characters in *S* and all possible substrings formed from the first *j* characters in *L*. Let *err(a,b)* = 1 if *a* = *b* and 0 otherwise.

Suppose *S*(*i*) = *D*(*j*). Then, *D*(*i*, *j*) = *D*(*i*-1, *j* -1).

Otherwise, we can compute *D*(*i*, *j*) as the smallest of scores computed from one of three actions:

(a) Substituting *S*(*i*) in *L*(*j*) with a penalty of 1 added to *D*(*i*-1, *j* -1).

(b) Deleting the *L*(*j*)th character, so that we are matching the first *i* characters of *S* with the first *j*-1 characters of *L*, which costs *D*(*i*, *j*-1) + a penalty of 1.

(c) Inserting a character at the *j*th position in *L* to make it match the character at the *i*th position of *S*. This costs the same as matching the first *i-1* characters in *S* with the first *j* characters in *L*, i.e., *D*(*i*-1, *j*) plus a added penalty of 1.

We can rewrite this as:

If *S*(*i*) = *D*(*j*) then *D*(*i*, *j*) = *D*(*i*-1, *j* -1)
else
*D*(*i*, *j*) = min( *D*(*i*-1, *j* -1) +1, *D*(*i*, *j*-1) +1, *D*(*i*-1, *j*) +1)

Note that in all cases, *D*(*i*, *j*) depends on a smaller index of either *i* or *j* or both, which creates an optimal substructure with reusable results.

If we start with *i* = 1, *j*=1, we can memoize the |*L*||*S*| entries and compute scores in time proportional to |*L*||*S*|. We set *D*(*i*, 0) = *i* and *D*(0, *j*) = *j* as boundary conditions. The string associated with each memoized position is the best match for that position, and is kept track of in the table depending on which of the three actions above were chosen to compute that position.

### 8 Lagrangian optimization

Both functions are continuous and twice-differentiable. We define the Lagrangian $F(x, y, \lambda) = x^3 + 2y + \lambda(x^2 + y^2 - 1)$. Setting $\nabla F = 0$, we get

$$\frac{\partial F}{\partial x} = 3x^2 + 2\lambda x = 0 \qquad \textbf{(EQ 1)}$$

$$\frac{\partial F}{\partial y} = 2 + 2\lambda y = 0 \qquad \textbf{(EQ 2)}$$

$$\frac{\partial F}{\partial \lambda} = x^2 + y^2 - 1 = 0 \qquad \textbf{(EQ 3)}$$

Solving (1), we get two solutions for $x$, denoted $x_1$ and $x_2$ :

$$x_1 = 0, x_2 = -\frac{2\lambda}{3}$$

Corresponding to $x_1$ we solve (3) to find $y_{11} = 1$, $y_{12} = -1$ and put these in (2) to get $\lambda_{11} = -1$, $\lambda_{12} = 1$. The extermal values of $z = x^3 + 2y$ for this solution of $x$ therefore are 2 and -2, achieved at the points (0,1) and (0,-1). Corresponding to $x_2$ we find from (3) that $\frac{4}{9}\lambda^2 + y^2 = 1$. Substituting $\lambda = -\frac{1}{y}$ from (2) and solving for $y$, we find that $y$ is complex, so that there are no real points ($x$,$y$) satisfying (3). Therefore, the only viable extremal points are the two found above, which correspond to a constrained maximum and constrained minimum respectively.

**9 Hill climbing**

We start with $K$ random points and compute the optimal value reached at each point. If we have $K$ unique results, we return the best point. Otherwise, we eliminate the repeated results, say $r$ of them, and start again with $r$ points and repeat the process (remembering those results already computed). When we reach $K$ different points the algorithm finishes and returns the global optimum. Note that we could iterate infinitely before finding the K local optima. However, without making any additional assumptions about the space, we cannot guarantees a better method to find the global optimum.

## *A1.5 Transform domain techniques*

## *A1.6 Queueing theory*

**10 Little's law**

(a) The mean waiting time is 180 min, and the arrival rate is 0.2 patients/minute. Thus, the mean number of patients is their product = 180*0.2 = 36. (b) We do not have enough information to determine the maximum size of the waiting room! We know we need at least 36 spaces, but it's possible that a burst of a hundred patients may arrive, for example, due to an incident of mass food poisoning. But, as a rule of thumb, some small integer multiple of the mean, such as three or four times the mean, ought to be enough. In real life, we are forced to work with such 'fudge factors' because it is often too difficult or too expensive to determine the exact arrival process, which, in any case, may abruptly change over time.

**11 A stochastic process**

At time 0, P[$X_0$=10] = 1.0.

At time 1, P[$X_1$ = 9] = 0.2; P[$X_1$ = 10] = 0.6; P[$X_1$ = 11] = 0.2.

At time 2, P[$X_2$ = 8] = 0.2(0.2) = 0.04; P[$X_2$ = 9] = 0.2(0.6) + 0.6(0.2) = 0.24; P[$X_2$ = 10] = 0.2(0.2) + 0.6(0.6) + 0.2 (0.2) = 0.44, and, by symmetry, P[$X_2$ = 11] = 0.24; P[$X_2$ = 12] = 0.04.

**12 Markov process**

The process is Markovian, because the probability of moving from stair $i$ to stairs $i$-1, $i$, and $i$+1 do not depend on how the person reached stair $i$.

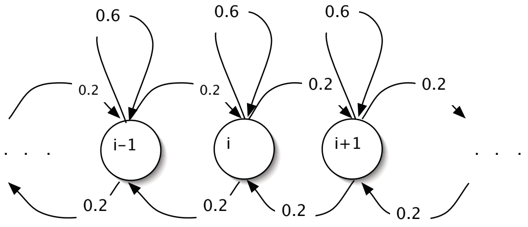
**13 Homogeneity**

The transition probabilities are time-independent, and therefore the process is homogeneous.

**14 Representation**

(a)

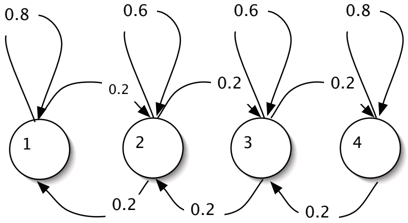
$$\begin{bmatrix} \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0.2 & 0.6 & 0.2 & 0 & \ldots & \ldots \\ \ldots & 0 & 0.2 & 0.6 & 0.2 & 0 & \ldots \\ \ldots & \ldots & 0 & 0.2 & 0.6 & 0.2 & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \end{bmatrix}$$



(b) The rows need to sum to 1, because at each time step, the process has to move to *some* state. The columns do not need to sum to 1 (think of a star-shaped state transition diagram with *N* states surrounding state 0, where state 0 has 1/*N* probability of going to any other state, and every state returns to state 0 with probability 1).

(c) We need to assume the boundary conditions. Suppose the at stair 1, the probability of staying at the same stair is 0.8, and at stair 4, the probability of staying at the same stair is also 0.8. Then, the transition matrix and state transition diagram are as shown below.

$$\begin{bmatrix} 0.8 & 0.2 & 0 & 0 \\ 0.2 & 0.6 & 0.2 & 0 \\ 0 & 0.2 & 0.6 & 0.2 \\ 0 & 0 & 0.2 & 0.8 \end{bmatrix}$$



## 15 Reducibility

The chain is irreducible because every state can be reached from every other state.

## 16 Recurrence

State 1 is recurrent because the chain is finite and irreducible. $f_1^1$ is the probability that the process first returns to state 1 after one time step, and this is clearly 0.8. $f_1^2$ is the probability that the process first returns to state 1 after two time steps, and this is 0.2 * 0.2 = 0.04. $f_1^3$ is the probability that the process first returns to state 1 after three time steps. This can happen after a transition to state 2, a self loop in state 2, and then back. Thus, the value is 0.2*0.6*0.2 = 0.024.

## 17 Periodicity

The chain is not periodic because of the self-loop in every state. A trivial chain with period *N* is a ring with *N* states, with the transition probability of going from state *i* to state *(i+1)* mod *N* = 1.

## 18 Ergodicity

No state in the chain is non-ergodic because the chain is finite aperiodic and irreducible.

## 19 Stationary probability

From Theorem 2, because the chain is ergodic, we obtain:

$$\begin{aligned}\pi_1 &= 0.8\pi_1 + 0.2\pi_2\\ \pi_2 &= 0.2\pi_1 + 0.6\pi_2 + 0.2\pi_3\\ \pi_3 &= 0.2\pi_2 + 0.6\pi_3 + 0.2\pi_4\\ \pi_4 &= 0.2\pi_3 + 0.8\pi_4\\ 1 &= \pi_1 + \pi_2 + \pi_3 + \pi_4\end{aligned}$$

This can be easily solved to obtain $\pi_1$=$\pi_2$= $\pi_3$=$\pi_4$= 0.25. (If you choose other assumptions for the boundary states, your computation will differ).

### 20 Residence times

$p_{11} = p_{44}$ = 0.8, so the residence times in these states is 1/(1-0.8) = 1/0.2 = 5. $p_{22} = p_{33}$ = 0.6, so the residence times in these states is 1/0.4 = 2.5.

### 21 Stationary probability of a birth-death-process

(a) Similarities: both are graphs with each node corresponding to a discrete state. Differences: the notation on an edge is the transition rate, not transition probability. The sum of rates leaving a node does not add up to 1, but total ingress rate matches total egress rate at each node.

(b) $\begin{bmatrix} -2 & 2 & 0 & 0\\ 2 & -6 & 4 & 0\\ 0 & 4 & -6 & 2\\ 0 & 0 & 2 & -2\end{bmatrix}$

(c) We have:

$-2P_0 + 2P_1 = 0$

$2P_0 - 6P_1 + 4P_2 = 0$

$4P_1 - 6P_2 + 2P_3 = 0$

$2P_2 - 2P_3 = 0$

$P_0 + P_1 + P_2 + P_3 = 1$

This yields: $P_0 = P_1 = P_2 = P_3 = 0.25$.

### 22 Poisson process

Consider a pure-death process, i.e. a birth-death process whose birth rates are zero. Clearly, the inter-departure times are nothing more than the residence times in each state. But we know that the residence times in a homogeneous continuous-time Markov chain are exponentially distributed (see Section 6.3.2 on page 121). QED.

### 23 Stationary probabilities of a birth-death process

We see that in this chain, $\lambda_i = \mu_{i+1}$ so immediately we get $P_0 = P_1 = P_2 = P_3$. By summing them to 1, we can see that they are all 0.25.

### 24 M/M/1 queue

It is not M/M/1 because the state-transition rates are state-dependent.

### 25 M/M/1 queue

(a) The packet length is 250 bytes = 2,000 bits, so that the link service rate of 1,000,000 bits/sec = 500 packets/sec. Therefore, the utilization is 450/500 = 0.9. When the link queue has 1 packet, it is in state $j$=2, because one packet is being served at that time. Thus, we need $P_2 = 0.9^2 * 0.1 = 0.081$. For the queue having two packets, we compute $P_3 = 0.9^3 * 0.1 = 0.0729$. For 10 packets in the queue, we compute $P_{11} = 0.9^{11} * 0.1 = 0.031$. (Compare these with values in Example 20 where the load is 0.8).

(b) The mean number of packets in the system is 0.9/1-0.9 = 9. Of these, 8 are expected to be in the queue.

(c) The mean waiting time is (1/500)/(1-0.9) = 0.002/0.1 = 0.02 s = 20 milliseconds.

### 26 Responsive (M/M/$\infty$) server

The ratio is:

$$\frac{e^{-\rho}\rho^j\frac{1}{j!}}{\rho^j(1-\rho)} = \frac{e^{-\rho}}{j!(1-\rho)} = \frac{1}{j!(1-\rho)e^{\rho}} = \frac{C}{j!}$$

, where *C* is a constant with respect to *j*. Therefore, for an M/M/$\infty$ queue the probability of being in state *j* diminishes proportional to *j!* compared to being in state *j* for an M/M/1 queue. Clearly, this favors much lower queue lengths for the M/M/$\infty$ queue.

### 27 M/M/1/K server

Packet losses happen when there is an arrival and the system is in state *j*=11. This is upper bounded by $P_{11}$, which is given by

$$P_{11} = \frac{1-\rho}{1-\rho^{K+1}}\rho^j = \frac{0.1}{1-0.9^{12}}0.9^{11} = 0.0437 \quad .$$

### 28 M/D/1 queue

(a) The mean number of customers in the system for such a queue is given by

$$\rho + \frac{\rho^2}{2(1-\rho)} = 0.9 + \frac{0.9^2}{2(0.1)} = 4.95$$

, which is roughly half the size of an equivalently loaded M/M/1 queue (from Exercise 17).

(b) The ratio is $\dfrac{\rho + \dfrac{\rho^2}{2(1-\rho)}}{\dfrac{\rho}{1-\rho}} = 1 - \dfrac{\rho}{2}$ . This tends to 0.5 as the utilization tends to 1.

(c) Under heavy loads, the mean waiting time for an M/D/1 queue is half that of a similarly loaded M/M/1 queue.

## *A1.7 Game theory*

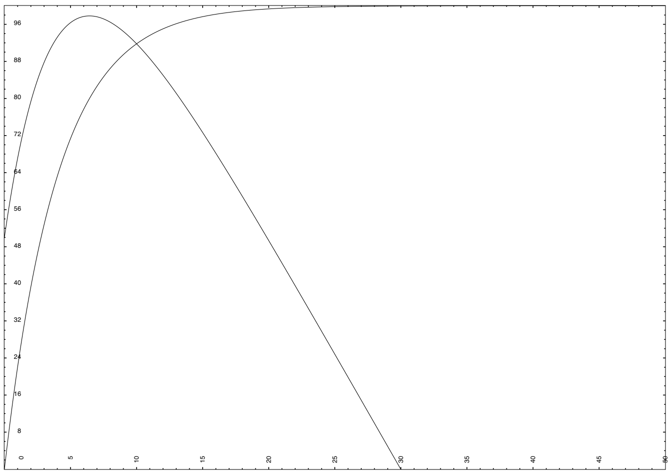
### 1 Preferences

Denote apple = A, banana = B, carrot = C, peach = P. We are free to choose utilities as we wish, so let U(A)=0, U(C) = 1. Then, U(B) =.7 and U(P) =.9, so you prefer peaches to bananas. (b) Let P(win B) = *p*. Then, .7*p* + 1(1-*p*) =.9, so .3*p* = .1, so *p* = 0.33.

### 2 Utility functions

Your net utility from transferring *x* GB is 100(1-$e^{-0.25x}$) if *x*<10 and 100(1-$e^{-0.25x}$) - 5(*x*-10) otherwise. The plot of these two functions is shown below:

It is clear that the maximum occurs at x=10 for a value of approximately 92. So, your utility is maximized by transferring exactly 10GB/month.



## 3 Pure and mixed strategies

The only possible first actions are: play corner, play middle, and play center. Depending on which move is played, the second player would have response, and depending on that response the first player would have a response etc. A pure strategy for each player is each valid response to the prior move (whether or not it is rational). A mixed strategy would play one of the pure strategies (i.e the entire sequence) with some probability. It turns out that in tic-tac-toe, with two expert players, a tie is guaranteed with a pure strategy, but a mixed strategy (depending over what you mix) could lose when played against an optimal strategy. So, it never makes sense to mix. In general, every component of a mixed strategy must be a potentially winning strategy. Otherwise, the mixed strategy would improve by discarding a component that can never win.

## 4 Zero-sum game

No, because utilities are only unique to a affine transformation.

## 5 Representation

| | L | M | H |
|---|---|---|---|
| Y | (1,a-1) | (2,a-2) | (3,a-3) |
| N | (0,0) | (0,0) | (0,0) |

## 6 Representation

We need to prove two things (a) if information sets are permitted every normal form game can be represented in extensive form (b) if information sets are permitted every extensive-form game can be represented in normal form. To prove (a): given a normal form game with *n* players, simply draw a tree of depth *n*, where all moves by the first player are associated with a node with an edge leading from the root to that node, and all nodes are in the same information set. Then, from each such node, draw an edge for each possible move for the second player, and place each set of nodes in the same information set. Repeat for each successive player, and label the leaves with the payoff from the corresponding array element. To prove (b): given the extensive form game, form paths from the root to each leaf. Decompose the path into moves by each of the players and find all possible moves by each player each time it is allowed to make a move. Let $S_i^t$ denote the set of moves that player *i* can move on its *t* turn. Then the strategy space for player *i* is the cross product of these sets. Finally, the normal form is an *n*-dimensional matrix with the *i*th dimension indexed by the strategy space of the *i*th player, and the corresponding element having the payoff for these strategies.

## 7 Best response

The best response depends on the value of *a*. For each of the strategies of the ISP, i.e., L, M, and H, the best response is Y if *a-price* > 0, otherwise it is N.

### 8 Dominant strategy

If you attend, your payoff is your utility for either pass or fail, but if you miss, your payoff is your utility for fail. Assuming that utility(pass) > utility(fail), your payoff for attending is as good as or better than the payoff for not attending. So, your dominant strategy is to attend.

### 9 Bayesian game

It is easy to verify that no matter the type of the Column player (strong or weak signal), the best response for Row if Column plays S is D and if Column plays D is S. Therefore, knowing the type of the Column player does not help Row, and the game does not have a dominant strategy for Row.

### 10 Repeated game

The one shot payoff is -3 for each, so the repeated payoff is -3* $\sum_{i=0}^{\infty} 0.6^i$ = -3/.4 = -7.5.

### 11 Dominant strategy equilibrium

It is dominant for both players to send rather than wait. In equilibrium, they always send right away so their packets always collide, and in fact, no progress is made, so that delays are actually infinite. This game illustrates the aphorism: haste makes waste. The EDCA protocol allows higher priority (delay sensitive) stations to wait for a shorter time than lower-priority stations before accessing the medium, therefore making it more probable that they would get access to medium and experience a shorter delay.

### 12 Iterated deletion

Consider the following game, where we only show the payoffs for Row:

| | C1 | C2 |
|---|---|---|
| R1 | 0 | 0 |
| R2 | 1 | -1 |
| R3 | -2 | 2 |

Neither R2 nor R3 dominate R1. However any mixed strategy of R2 and R3 that plays R3 with a probability greater than 2/3 dominates R1. Therefore, we can delete R1 from the game.

### 13 Maximin

In Example 10, Row can get as low as -1 with S, but at least 0 with D, so its maximin strategy is D. Column is assured 1 with S, so its maximin strategy is S, and the equilibrium is DS.

In Example 14, Row maximizes its minimum payoff with S. The game is symmetric, so the maximin equilibrium is SS.

### 14 Maximin in a zero-sum game

In Figure 3, note that when $p$ is smaller than 0.5, the column player can play pure strategy C1 to reduce Row's payoff below 2.5. Similarly, if $p$ is greater than 0.5, Column can use a pure strategy C2 to reduce Row's payoff. For any value of $p$, Column can play a mixture $q$C1 + (1-$q$) C2 to give Row a payoff of $q(p+2) + (1-q)(4-3p)$. To make this smaller than 2.5, we set $q(p+2) + (1-q)(4-3p) < 2.5$, or $q > (3-6p)/(4-8p)$. For instance, if $p$=0, $q > 3/4$, and if $p$=1, $q$>3/4. (The inequality is not valid when $p$= 0.5.)

### 15 Nash equilibrium

Let the row player play $p$H + (1-$p$)T. Then, its payoff, given Column's mixed strategy, is $p(q-(1-q))+(1-p)(-q+(1-q))$ = $4pq -2q -2p +1 =(1-2p)(1-2q)$. If $q < 0.5$, $p$ should be 0, otherwise $p$ should be 1. Intuitively, if the Column player is more likely to play T, then Row should play T for sure and *vice versa*.

### 16 Correlated equilibrium

Consider an external agency that tells the players to play $p$DS + (1-$p$)SD. When Row is told to play D, it knows that it will get a payoff of -1 if it deviates. Similarly, when told to play S, it will get 0 if it deviates (instead of 1). So, it will not deviate, independent of the value of $p$. By symmetry, the same analysis holds for Column, and therefore we have a correlated equilibrium. The external agency can arrange for any desired payoffs to Row and Column by adjusting $p$.

## 17 Price discrimination

Assume that the valuations of each player are $v_1,...,v_n$ for minimum quantities of $q_1,...,q_n$. The scheme is essentially to charge $v_i$ for $q_i$ adjusting for the fact that player $i$ could buy multiples of $q_j$ $j<i$ if that minimizes its total cost.

## 18 VCG mechanism

(a) The overall function is $(20+40+80)(1-e^{-0.5x}) - 20x = 140(1-e^{-0.5x}) - 20x$.

(b) The types are the only unknowns in the utility functions, i.e. 20, 40, and 80 respectively.

(c) The optimal social choice comes from maximizing the function in (a). Setting $f(x) = 140(1-e^{-0.5x}) - 20x$, solve for $f'(x^*)=0$, so that $x^* = 2.5055$.

(d) To compute $x^{-1}$, we maximize $(120(1-e^{-0.5x}) - 20x)$ to get 2.197. Similarly, $x^{-2} = 1.832$, and $x^{-3} = 0.8109$. Thus, $p_1 = v_2(x^{-1}) + v_3(x^{-1}) - (v_2(x^*) + v_3(x^*)) = (40+80)(1-e^{-0.5*2.197}) - (40+80)(1-e^{-0.5*2.5055}) = 120*(e^{-1.25275-}e^{-1.0985})$ $= -5.718$.

Similarly, $p_2 = v_1(x^{-2}) + v_3(x^{-2}) - (v_1(x^*) + v_3(x^*)) = 100(e^{-1.25275-}e^{-0.5*1.832}) = -11.439$,

$p_3 = v_1(x^{-3}) + v_2(x^{-3}) - (v_1(x^*) + v_2(x^*)) = 60(e^{-0.5*0.8109}-e^{-1.25275})= 60*(e^{-1.25275-}e^{-0.4055}) = -22.857$.

(e) No, the budget is not balanced: the CIO has to pay each department.